TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program : B2646

Studijní obor : 1802R007

Analýza záznamu měření parametrů spotřeby elektrické energie

Analysis of the record parameters measuring the power consumption

Bakalářská práce

Autor: **Martin Pěnička**

Vedoucí práce: Ing. Jan Kraus

Konzultant: Ing. Pavel Štěpán

V Liberci dne 17.5.2012

Fakulta mechatroniky, informatiky a mezioborových studií

Zadání bakalářské / diplomové práce

Příjmení a jméno studenta (osobní číslo - nepovinné)	Martin Pěnička (M09000165)
Zkratka pracoviště	MTI
Datum zadání BP/DP	14. 10. 2011
Plánované datum odevzdání	18. 5. 2012
Rozsah grafických prací	dle potřeby dokumentace
Rozsah průvodní zprávy	
Název BP/DP (česky)	Analýza záznamu měření parametrů spotřeby elektrické energie
Název BP/DP (anglicky)	Analysis of the record parameters measuring the power consumption

Zásady pro vypracování BP/DP (text nijak neformátujte, pouze očíslujte jednotlivé body .. 1) ... 2) ... atd. a každý bod uveďte jako nový odstavec)**:**

1) Seznamte se s možnostmi analyzátoru SMPQ a s obsahem archivu měření tohoto přístroje.
2) Vyberte vhodný nástroj pro vývoj automatizovaných analýz souboru měření.
3) Zaměřte se na statistické vyhodnocení časových řad hodnot měřených veličin, vyhledávání podobností v souboru a clustering.
4) Implementujte navržené postupy a na vhodných datech demonstrujte jejich funkčnost.

Prohlášení

Byl(a) jsem seznámen(a) s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu TUL.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Diplomovou práci jsem vypracoval(a) samostatně s použitím uvedené literatury a na základě konzultací s vedoucím diplomové práce a konzultantem.

Datum 17.5.2012

Podpis

Poděkování

Rád bych poděkoval svému vedoucímu Bakalářské práce, panu Ing. Janu Krausovy za cenné rady při řešení problémů spjatých s prací a za podporu při psaní práce. Dále bych chtěl poděkovat společnosti KMB systems za možnost pracovat na reálný datech z provozu, které by byli jinak těžko dostupné.

Abstrakt

Inteligentní zpracování a třídění rozsáhlých databází je dnes běžnou praxí zejména ve velkých nadnárodních společnostech a v akademické sféře. Avšak provádění algoritmů pro dobývání znalostí na prostých časových sériích není tak rozšířenou doménou, a v oblasti zpracování údajů z měření parametrů spotřeby elektrické energie se nesetkáme s běžným použitím. Cílem této práce je seznámit se s algoritmy umožňujícími takovéto manipulace s daty, seznámení se s existujícími přístroji určenými k měření parametrů spotřeby elektrické energie a archivem takového měření. Dale je nutné nalézt vhodný nástroj pro implementaci těchto metod, programovací jazyk či matematické vývojové prostředí. Hlavním cílem je implementace všech potřebných metod a jejich uvedení na reálných datech do praxe a vytvoření nástroje, který bude tyto metody implementovat.

Klíčová slova

paramery elektrické energie
timeserie
datamining
clustering
python

Abstract

Inteligent processing and classification of very large database is nowadays common matter especially in huge global companies and in academical spheres. However, performing algorithms for datamining knowledge on simple time series isn't wide domain, and we don't meet common usage in area of processing data from mesuring power consumption parameters neither. The aim of this work is to familiarize with algorithms, which can enable this kind of manipulation with data, to present existing instruments designated for measuring power consumption parameters and to make familiar with archieves of these kinds of measuring. Tne next necessary thing is to find a suitable instrument for implementation of these methods, programming language or mathematic IDE. The main target is to implement all these needed methods and it's set on real data into practise and creating instrument which would implement all these methods.

Keywords

parametres of the power consumption
timeserie
datamining
clustering
python

Obsah

Seznam ilustrací

Seznam tabulek

1 Úvod

Metody pro automatizovanou analýzu databází a clustering jsou v současné době rozšířeny v mnoha oborech lidské činnosti. Společně s růstem výkonu počítačů, diskových prostor a dalších faktorů se tyto algoritmy dále rozšiřují do dalších oborů a je možné rozšířit je do pracovišť,které nedisponují nejvýkonnějšími stroji. Většina z těchto algoritmů je výpočetně a časově velice náročná a proto nebylo dříve myslitelné snadno a rychle je testovat na obyčejných domácích počítačích. Jako příklad lze vzít v potaz, že průběh samotného filtrování ještě před samotným započetím výpočtu algoritmu může při velkém rozsahu dat i nyní trvat téměř dvacet minut. Pro pochopení této problematiky a poznání aplikace těchto algoritmů, se první kapitola zabývá prezentací různých oborů lidské činnosti, kde clustering a pokročilá automatická analýza dat nachází své místo. Je nutné seznámit se nejen se samotnou aplikací algoritmů, ale také s důvody, které k jejich nasazení vedli, pro lepší pochopení snahy jejich nasazení do oblasti analýzy záznamu měření parametrů spotřeby elektrické energie. V této oblasti jsou velké možnosti jeho nasazení. Zejména tam, kde se práci, kterou některá z těchto metod zvládne vypočítat během krátké doby, musí nyní vykonávat zdlouhavě člověk. Také umožní naprosto jiný pohled na naměřená data. Dalším krokem je seznámení se s přístroji, které tyto data poskytují. V další části první kapitoly je prezentován měřící přístroj SMPQ, vyvinutý společností KMB systems[1]. Díky přístupu k tomuto archivu měření je v celé práci pracováno s reálnými daty zaznamenanými na různých místech nasazení, elektrárnách, firmách s elektrickými stroji ale i v obyčejných domácnostech. V poslední části první kapitoly je popsán archiv měření, se jehož daty se v této práci pracuje. Následujícím krokem bylo vybrání vhodného programátorského nástroje pro implementaci těchto algoritmů. Ze současného nepřeberného množství programovacích jazyků a matematických vývojových prostředí bylo vybráno několik nejvhodnějších pro porovnání jejich možností. Druhá kapitola se proto zabývá výběrem tohoto nástroje, prezentací jejich schopností a následným zdůvodněním výběru konečného nástroje. Dodaná data měření nabývají enormních rozměrů při analýze intervalů v řádu měsíců. Proto je nutné načtená data optimalizovat a vybrat z nich pouze relevantní části. Z existujících filtračních technik je několik prezentováno v třetí kapitole, nachází se zde několik grafů zobrazující jejich práci a také tabulky porovnávající jejich výkon. Poslední částí je popis možné metody pro výběr vhodných vlastností pro reprezentaci vzorků souboru měření. Samotný obor problematiky automatizované analýzy dat a clusteringu je poměrně obsáhlý a dá se rozdělit do několika tříd. Bylo vybráno několik algoritmů, každý odlišně pracující pro prezentaci jejich funkce a výsledků. V kapitole číslo pět je podrobně rozebráno několik z těchto algoritmů a jsou prezentovány výsledky

jejich práce na reálných souborech měření. Je zde také vzájemné porovnání jejich výsledků a výkonu v závislosti na čase nutném k jejich výpočtu. Pro účely prezentace výsledků této práce ale také pro možnosti dalšího testování byla vytvořena aplikace umožňující provádět požadované operace na nahraných datech. Další grafy zobrazující závislost výsledků clusteringu a filtrace se nacházejí v příloze. V kapitole šest je popsáno její ovládání i možné funkce. Také jsou zde popsány jednotlivé fáze samotného uvádění algoritmu do praxe, nutně implementované metody a použité knihovny.

2 Analýza záznamu měření parametrů elektrické energie

2.1 Prostředky a aktuální stav

Liberecká společnost KMB systems se zabývá výrobou hardwarových přístrojů pro měření různých parametrů elektrické energie a také komplexních softwarových řešení pro analýzu dat zaznamenaných těmito přístroji. Pro tuto práci je důležitý zejména měřící přístroj SMPQ. Jeho podrobnější popis se nachází dále v této kapitole. Přístroje tohoto typu mají široké možnosti použití. Používají je dodavatelé elektrické energie, například provozovatelé solárních elektráren pro měření efektivity a výkonu jednotlivých bloků elektrárny. Jsou vhodné do velkých moderních inteligentních budov s pokročilou správou spotřeby a její následné optimalizace dle výsledků získaných z archivů měření. Měřící přístroje jsou využívány k měření spotřeby elektrických strojů a následnému odhalování chyb pro optimalizaci jejich funkce. V blízké budoucnosti bude běžné jejich použití k měření parametrů spotřeby elektrické energie také v rodinných domech a v libovolném prostředí, kde je třeba zaznamenávat a analyzovat elektrickou energii procházející vedením. V archivech měření z těchto přístrojů se nachází rozsáhlé množství souborů měření z různých výše zmíněných zdrojů. Díky spolupráci s firmou KMB je možné na těchto datech dále provádět experimentální funkce typu porovnávání podobností a clustering, které nepatří mezi funkcionalitu softwarových produktů poskytovaných touto společností. Účel této analýzy se liší podle uživatele přístrojů. Pro velkého dodavatele elektrické energie může sloužit například pro cílený prodej elektrických tarifů. Umožňují rozdělit zákazníky společnosti do kategorií podle průběhu jejich denní spotřeby. Zákazníkům s vysokou noční spotřebou je možné poté cíleně nabízet prodej tarifů pro noční proud. Pro zákazníky používající přístroje pro hlídání spotřeby v závodech na elektrických strojích je výhodné rychle vyhledávat v datových záznamech nestandardní průběhy výkonu či spotřeby strojů a aktivně vyhledávat možné závady na strojích či rušení na vedení. Použití v inteligentních i klasických budovách umožňuje aktivně sledovat spotřebu a vyhledávat nesrovnalosti. Jako příklad lze uvést nalezení poruchy málo používaného přístroje, která se vyznačuje vysoce zvýšenou spotřebou. Tato porucha by se vyznačovala nestandardním zvýšením spotřeby v určitých dnech měření, které by se podařilo díky těmto postupům velice rychle nalézt. Poslední příklad použití je společný pro všechny uživatele tohoto přístroje. Jedná se o optimalizaci údajů v databázi. Při běžném provozu se očekává, že profily spotřeby či výkonu budou stále stejné a odchylky se budou projevovat pouze při chybách. Databáze se díky tomu plní téměř duplicitními údaji. Pomocí těchto metod je možné vyfiltrovat pouze nestandardní a tudíž podstatné údaje

a ostatní údaje neukládat, nebo ukládat pouze jejich průměr jako ukázku průměrného dne. Pomocí této metody lze dosáhnout obrovské úspory místa v databázi. Cílem této práce je proto nalézt a implementovat metody, které umožňují tyto operace na datech. Pro vývoj a testování budou používána reálně naměřená data z archivů měření přístroje SMPQ. Budou použity inteligentní algoritmy patřící do rodiny algoritmů pro dobývání znalostí (data mining) a strojové učení (machine learning). Tyto rodiny algoritmů patří do poměrně rozsáhlého oboru umělé inteligence, které se postupně s jejich zvyšující rozsáhlostí a složitostí vydělili do vlastního podoboru. U všech těchto algoritmů se počítá s vysokým stupněm automatizace a vlastního získávání informací bez zásahu lidského faktoru. Dále se také předpokládá jejich katalogizace a snaha pochopit podstatu dat. Část těchto algoritmů používaných pro clusterovací operace lze rozdělit do dvou skupin. Jsou to skupiny „supervised learning“ a „unsupervised learning“. Názvy nemají oficiální české ekvivalenty a daly by se přeložit jako „učení pod dohledem“ a „učení bez dohledu“. Při použití algoritmů pro učení pod dohledem se předpokládá vstupní množina dat, která je již rozdělena do různých skupin, Algoritmus se naučí podobnosti, které panují mezi členy skupiny a je připraven jakákoli následující samostatná data, která přijdou na vstupu, zařadit do některé skupiny. Vytvoří pro každou skupinu její charakteristiku a profil, podle které jsou následující data porovnávána. Tato skupiny algoritmů je vhodná pro on-line vstup dat. Nicméně jejich použití je vázáno předchozí znalostí struktur, které panují v celém datovém souboru. Algoritmy, patřící do skupiny pro učení bez dohledu, jsou vhodné pro rozřazování surových dat, kde nemáme žádnou představu o jejich struktuře a podobnostech jednotlivých souborů. Algoritmus porovnává jednotlivé vzorky měření mezi sebou podle předem zadaných parametrů a podle podobnosti sám najde struktury v datech bez zásahu lidského uživatele. Tyto algoritmy jsou poměrně výpočetně náročné pro velký rozsah dat. Oba typy algoritmů lze vhodně kombinovat. První proběhne učení bez dohledu a najde struktury v datech. Poté předá výsledky algoritmu pro učení pod dohledem, který se naučí rozeznávat jednotlivé skupiny a veškerá nově příchozí data jsou již zařazována do existujících skupin. Tyto algoritmy jsou poměrně dlouhou dobu známé (například v práci používaný algoritmus K-means byl publikován počátkem osmdesátých let), ale existují pouze obecné implementace, které se musí dodatečně upravit a optimalizovat podle použitých dat. Jejich velké rozšíření do všech oblastí proběhlo až v posledních letech, zejména kvůli dřívější drahé diskové kapacitě a nedostatečnému výkonu počítačů, kdy získání výsledků pomocí tohoto algoritmu trvalo neúměrně dlouhou dobu. V současnosti se objevila velká spousta pokročilých a inteligentnějších algoritmů majících použití v širokém spektru dobývání znalostí z různých databází. Jako příklad lze uvést používání clusterovacích algoritmů pro výzkum rozdělení trhu[2]. Velké internetové obchody je používají zejména již k výše uvedené cílené reklamě. Rozdělují zákazníky do skupin dle jejich vkusu.

Členům těchto skupin nabízí dále formou reklamy zboží zakoupené a kladně ohodnocené ostatními členy skupiny. Díky tomuto postupu se velkou mírou zvyšuje pravděpodobnost úspěchu při prodeji produktů. Další možnost použití je procházení lékařských databází záznamů ve formě časových sérií, což je na příklad záznam EKG[3]. Jako příklad bych uvedl určování náchylnosti pacientů k infarktu podle standardním způsobem nezřetelných faktorů. Algoritmus ze skupiny učení pod dohledem projde záznamy EKG a záznamy dalších určených parametrů. Tak najde charakteristiku, která určuje budoucí náchylnost k infarktu a použije ji pro porovnání náchylnosti budoucích pacientů. Oblastí, kde jsou tyto metody v současnosti hojně používané, je oblast obchodu s cennými papíry a komoditami[4]. Porovnávají se společné vlastnosti akcií a komodit v závislosti na jejich parametrech, minulému růstu a demografických podmínkách. Z tohoto celku parametrů se algoritmus snaží vyčíst strukturu, rozdělit akcie do skupin a odhadovat budoucí růst či pokles. Na téma používání těchto algoritmů v oblasti energetiky lze nalézt také několik odborných prací, například zmíněné dělení zákazníků pro prodej tarifů[5], ale jejich použití v praxi v této oblasti není tak obvyklé jako ve výše zmíněných oblastech. Cílem je tedy navrhnout, otestovat a následně uvést v použitelnou praxi metody z tohoto oboru algoritmů v oblasti měření parametrů spotřeby elektrické energie ve formě časových řad. Toto je možné pouze díky spolupráci s firmou KMB systems a jejími přístroji.

2.2 Analyzátor SMPQ

SMPQ[6] je multifunkční měřící přístroj a analyzátor kvality elektrické energie vyvinutý společností KMB systems. Přístroj má vysokou přesnost měření a díky vysokokapacitní paměti umožňuje dlouhé kontinuální měření. Měří napětí na čtyřech vstupech a procházející proud také až na čtyřech vstupech. Umožňuje připojení třífázových sítí (tři až pět vodičů) a vyhodnocování harmonických a meziharmonických složek až do 50. řádu podle IEC 61000-4-7 ed. Přístroj vyhodnocuje průměrné veličiny podle několika metod, např. metodou pevného nebo plovoucího okna. Zaznamenává události jako přepětí, poklesy a výpadky napětí. Elektroměr tohoto přístroje umožňuje čtyřkvadrantní měření elektrické energie, jednofázové až třífázové hodnoty a záznam maxim průměrných činných výkonů za současný a předchozí měsíc. Přístroj má velké možnosti připojení a komunikace přes různé typy sběrnic. V základu je to USB, může být dále rozšířen o RS-232 nebo Ethernet. Data tak mohou být zaznamenávána do vestavěné paměti a při překročení kapacity archivována do databáze na serveru. V příloze se nachází obrázek přístroje SMPQ.

2.3 Obsah archivu měření

V archivu měření přístroje SMPQ jsou uchovány veškeré naměřené údaje[7]. Vedle záznamů skutečných dat jsou to především záznamy událostí jako přepětí nebo výpadky, záznamy nastavení přístroje a podobné uložené do logů, které mohou být pro pozdější analýzu taktéž užitečné. Archiv je rozdělen do několika archivních systémů podle druhu zaznamenaných údajů. Jsou to hlavní archiv, (S/M)-Profil, Archiv logů, PQ hlavní archiv, archiv PQ událostí, archiv PQ oscilogram, archiv PQ událostí – průběh, archiv odečtů elektroměru a archiv Pmax. V příloze se nachází několik ilustračních obrázku naměřených dat z tohoto archivu.

V hlavním archivu se nacházejí veškeré naměřené veličiny, měřené v předem nastaveném intervalu. Klasickou volbou je automatické měření všech veličin v intervalu jedné minuty. V archivech obsahujících (S/M)-Profil jsou uloženy nejdůležitější hodnoty (napětí, proudy, výkony, frekvence,..) z měření jednotlivého dne. S-Profil dne je profil, který byl zaznamenán v nastavený den. V M-Profilu jsou uloženy maximální součty všech proudů. V archivu logů událostí jsou zaznamenané vlastní události přístroje, jako jsou změny nastavení, výpadky proudu a resety. Tyto události mohou být důležité pro pozdější vyhodnocení naměřených dat. PQ hlavní archiv je samostatný archiv pro statistické vyhodnocení. V základním nastavení ukládá informace o vyhodnoceních z přístroje v intervalu deseti minut a jednoho týdne. Vyhodnocení jsou definována normou EN 50160 pro vyhodnocování kvality energie. V dalším nastavení mohou být intervaly změněny. Archiv PQ událostí obsahuje informace o napěťových událostech typu výpadků, přepětí a poklesů. Informace jsou uloženy ve formátu nejnižšího naměřeného napětí, počátečního a koncového času fáze. Informace uložené v tomto archivu jsou důležité pro pozdější vyhodnocení, protože důsledkem výše zmíněných napěťových událostí jsou zvýšené náklady nebo škody. Archiv PQ oscilogram je pevně spjat s archivem PQ událostí. Obsahuje detailní průběhy zaznamenaných napěťových událostí. Archivuje záznamy dlouhé 600ms, které začínají 200ms před zaznamenanou událostí. Vzorkovací frekvence záznamu je 32 vzorků za periodu. Přístroj SMPQ je vybaven čtyřkvadrantním, třítarifním elektroměrem. Archiv odečtů elektroměru obsahuje hodnoty naměřené tímto elektroměrem. Interval měření může být nastaven od nejnižší hodnoty patnácti minut až do maximální hodnoty periody jednoho týdne. Naměřené hodnoty jsou archivovány po kvadrantech, fázích nebo tarifech. V archivu Pmax je uloženo vždy pouze několik hodnot pro celý měsíc měření. Archiv obsahuje maximální hodnotu třífázového výkonu a přesný čas výskytu události.

3 Výběr vhodného programátorského nástroje

Pro práci na projektu bylo nejprve podstatné vybrat vhodný programátorský nástroj a vytvořit sadu základních podmínek pro výběr vhodného nástroje. Poté bylo nutné vybrat z velkého množství existujících programovacích jazyků i matematických prostředí nástroje splňující určené podmínky. U některých z těchto nástrojů byla otestována jejich schopnost provádět požadované operace. Nástroj na programování a testování algoritmů požadovaných vlastností by měl být nejlépe skriptovací interpretovaný programovací jazyk nebo matematické prostředí pro vědecké výpočty, používající skriptování stejné úrovně a to z důvodu rychlosti testů a nezávislosti na platformě. Dále by měl být především vysokoúrovňový, čímž bude odstraněna nutnost zabývat se řešením zbytečných problémů nesouvisejících s tématem projektu. Myšlena je především nutnost vlastní implementace potřebných datových struktur, jako jsou matice, vektory a potřebné funkce pro operace s nimi. Důležitá je knihovna funkcí k rychlému vykreslování grafů pro vizuální ověření a interpretaci získaných výsledků. Dále pak rozsáhlá knihovna matematických funkcí pro požadované výpočty. Veškerá dodaná data budou exportovaná do souborů formátu programu Microsoft Office Excel, proto je nutné nalézt nástroj s dobrou vnitřní podporou čtení tohoto formátu. Nástroj by měl mít také vlastní podporu pro analýzu časových sérií a statistiku, popřípadě její podporu ve formě dodatečné knihovny. Veškerý vývoj probíhal na platformě Linux, ovšem testování proběhlo jak na platformě Linux, tak i na platformě Microsoft Windows.

Všechny tyto vlastnosti v podstatě splňují tyto vybrané čtyři nástroje : R-Project, Mathworks Matlab, GNU Octave a programovací jazyk Python. V následující části kapitoly jsou tyto nástroje popsány a analyzovány. Jsou vyhodnoceny jejich pozitivní a negativní vlastnosti a zdůvodněno rozhodnutí pro konečný výběr nástroje. Pro tento nástroj jsou následně důkladně prezentovány jeho hlavní vlastnosti důležité pro řešení této práce a příklady jejich použití.

3.1 R-Project

R je vývojové prostředí a objektově orientovaný interpretovaný jazyk[8]. Je dostupné pro všechny nejrozšířenější platformy (Windows, Mac OS,. Linux) ve formě předkompilovaných balíků. Na ostatní platformy může být zkompilován ze zdrojového kód. Pro R bylo vytvořeno komunitou mnoho dalších vývojových prostředí, např. multiplatformní prostředí v jazyku Java a samostatný balík integrující podporu po R do programu Microsoft Excel. Licence pro R je GNU

GPL, vše je volně dostupné ze stránek projektu nebo z repositářů linuxových distribucí, celý projekt nemá podporu žádné komerční instituce a je celý vyvíjen samostatnou vývojářskou komunitou. R-Projekt má velkou podporu rozšiřitelnosti o nové funkce samotnými uživateli, proto je možné doinstalovat velké množství knihoven téměř pro jakoukoli funkci. Jazyk lze použít jako interpretovaný z příkazové řádky nebo v prostředí s grafickým uživatelským rozhraním. S objekty jazyka lze také manipulovat pomocí jazyka Java nebo C. R má vestavěnou dobrou podporu pro cíle této práce, zejména analýzu časových sérii a clustering dat. Podporuje spoustu potřebných datových struktur jako matice a vektory. Primární zaměření pro R je statistika a vykreslování grafů. V nástroji R bylo provedeno několik testů funkcí pro načítání datových souborů, vykreslování grafů a matematických funkcí.

3.2 Mathworks Matlab

Matlab je velice známý, v akademické i profesionální sféře často používaný produkt[9]. Je primárně určen pro numerickou matematiku a lineární algebru (odtud název Matlab – MAtrix LABoratory - maticová laboratoř). Hlavním cílem je prostředí pro rychlé řešení matematických problémů bez hlubší znalosti programování. Celý Matlab je vývojové prostředí s vlastním programovacím vysokoúrovňovým jazykem, který může být použit jak ve formě scriptů, tak i s pomocí příkazového interpreta. Obsahuje také nástroje pro tvorbu grafických uživatelských prostředí. Může komunikovat s programy napsanými v jiných jazycích, např. C/C++, Java a Fortran. Dále může být Matlab rozšířen o dodatečné balíky určené pro pokročilou simulaci, paralelní výpočty, zpracování signálů, ale také pro zpracování videa určeného pro počítačové vidění. Matlab je projekt s uzavřenou licencí a pro komerční použití je poměrně drahý. Přestože by byl na řešení zadaného problému vhodný, jeho finanční nedostupnost se jeví jako největší problém. Matlab neposkytuje žádné licence pro nekomerční nebo studijní použití ale pouze zkušební demo. Z tohoto důvodu byl nástroj Matlab vyřazen kvůli praktické nemožnosti jeho získání legální cestou. I přes jeho nesporné výhody nebyly provedeny ani testy jako na ostatních nástrojích. To to je ale výhoda pro komerční použití díky záruce společnosti Mathworks, která za Matlabem stojí.

3.3 GNU Octave

Octave je vysokoúrovňový programovací jazyk primárně určený pro numerické výpočty[10]. Projekt Octave je prakticky velice podobný projektu Matlab. Mnoho uživatelů ho dokonce považuje za open source alternativu k tomuto programu, ač tak nebyl koncipován. Octave je o několik let mladší než Matlab a poskytuje v podstatě stejnou funkcionalitu na základní úrovni. Přirozeně umožňuje výpočty z lineární algebry, integrace funkcí, řešení diferenciálních rovnic a manipulaci s nelineárními rovnicemi. Dodatečné vysokoúrovňové funkce poskytují dodatečné balíky, které nejsou součástí Octave jsou samostatně vyvíjeny komunitou. Jako příklad poslouží gnuplot určený k vykreslování grafů. Moduly mohou být psané v samotném jazyce Octave, ale také v jazyce C a dalších jazycích. Tento jazyk je interpretovaný a v základu se ovládá pouze přes příkazového interpreta. V Unixových systémech je přímo integrovatelný do emulátoru terminálu. Skripty lze zapisovat do souborů a přes interpreta je spouštět. Existuje i mnoho dodatečných programů poskytujících grafické uživatelské rozhraní. Jako příklad může být uveden program příhodně pojmenovaný GUI Octave. Programovací jazyk Octave je velice podobný tomu z nástroje Matlab a po dodržení určitých pravidel syntaxe, jsou skripty mezi nástroji téměř kompatibilní. Vzhledem ke své licenci GNU GPL, dodatečné rozšiřitelnosti a upravitelnosti, je Octave velice často používán v akademické sféře.

3.4 Programovací jazyk Python

Python je dynamický objektově orientovaný programovací vysokoúrovňový jazyk[11]. V základu se spouští v příkazovém interpretu, ale je vhodný i na psaní desktopových aplikací s grafickým uživatelským rozhraním. Jsou vytvořeny implementace pro jeho použití jako scriptovacího jazyka pro objektové jazyky Java (Jython) a C# (IronPython). V těchto implementacích lze použít grafické knihovny obou jazyků (Swing pro Javu, Windows Forms pro C#), v klasickém Pythonu například GTK nebo Qt. Python používá dynamické typování proměnných. Syntaxe jazyka má striktní podmínky na čistotu a přehlednost, která nutí programátora k udržování pořádku. Nedodržování těchto principů vede k chybnému kódu a interpret kód nepřeloží. Python má otevřenou licenci pro nekomerční i komerční použití. Jazyk je multiplatformní, linuxové distribuce ho mají ve své základní instalaci. Běží na systémech

Windows, Linux, Mac, OS/2 a další. Existuje také i verze pro telefony Nokia série 60. Stejný kód funguje beze změny na všech platformách. Jazyk má obrovské množství dodatečných knihoven, které se neustále rozvíjejí. Například knihovny NumPy (numerické výpočty), SciPy (vědecké výpočty) a Matplotlib (vykreslování grafů) se v určitých funkcích vyrovnají matematickým prostředím typu Matlab a proto ho na některých pracovištích začíná nahrazovat. Python je napsán v jazyce C, a tudíž se i knihovny pro jazyk dají psát v samotném Pythonu i v jazyce C. Moduly v C jsou o něco rychlejší než v Pythonu. Velice špatnou vlastností jazyka Python je nekompatibilita jednotlivých verzí. Vyvíjí se několik verzí jazyka současně, podle toho, jak jsou rozšířené (v současnosti nejvíce verze větve 2.7 a větve 3.2). Dostupné knihovny jsou taktéž vázané použitou verzí jazyka, proto je nutné vždy před započetím projektu dobře zvážit, které verze použít. Všechny výše zmíněné matematické knihovny jsou v současnosti vázané na verzi 2.7 ale pracuje se na jejich verzích kompatibilních s vyššími verzemi Pythonu. Za další negativum je možné považovat absenci klasických abstraktních tříd.

3.5 Porovnání nástrojů

V následující tabulce je zobrazeno porovnání nástrojů podle několika faktorů použitelnosti. Následný výběr nástroje však podléhá přísnějším podmínkám, které nelze v jednoduché tabulce naplno porovnat. Volná licence je pro tuto práci důležitá z důvodu přístupu k veškerým funkcím nástroje bez omezení, která vyžadují zkušební či omezené verze komerčních produktů (například požadavek neustálého připojení k internetu). Volné komerční použití v uvedeném kontextu je důležité pro budoucí eventuální začlenění do již existujících softwarových řešení. Vzhledem k tomu že veškerý vývoj projektu probíhal na platformě Linux a a testování na platformě Windows, je nutnost multiplatformního použití zřejmá. Kvalitní možnost integrace je nutností zejména kvůli možnosti integrace do již zmíněného existujícího produktu. Použití metod, které budou výsledkem této práce bez nutnosti jejich přepisu do jiného jazyka, použitého pro vývoj softwarového produktu, a použití vybraného jazyka jako skriptovacího nástroje pro tento produkt, jsou velkou budoucí výhodou. Cena zvolených nástrojů je zde zmíněna pouze pro porovnání.

	Licence	Volné kom. použití*	Vlastní rozšiřitelnost **	Multiplatformní	Možnost integrace***	Cena
R-Project	GNU GPL	ne	ano	ano	ne	0,-
Matlab	uzavřená	ano	ne	ano	ne	59 980,- (komerční)
GNU Octave	GNU GPL	ne	ano	ano	ne	0,-
Python	Otevřená pro vše	ano	ano	ano	ano	0,-

Tabulka 1: Porovnání vlastností programátorských nástrojů

* Volné komerční použití v tomto kontextu znamená naprostou volnost použití k jakýmkoli účelům. Podmínky striktní GPL licence požadují uvolnění veškerých zdrojových kódu pod stejnou licencí pro projekty postavené nad těmito projekty.

** Jako vlastní rozšiřitelnost je zvolena možnost jakéhokoli vlastního zásahu do samotného systému (jako nutná podmínka je otevřenost zdrojových kódů). Dodatečná rozšiřitelnost tvorbou knihoven je samozřejmá vlastnost téměř každého programovacího jazyka.

*** Možností integrace se rozumí jako snadná použitelnost systému jako skriptovacího jazyka jako nějaký jiný programovací jazyk či softwarový systém bez dodatečné nutnosti tvorby vlastních modulů.

3.6 Vybraný nástroj a důvody jeho použití

Všechny výše uvedené technologie jsou více méně vhodné pro zpracování tématu této práce. R-Project má sice rozsáhlé a vyhovující možnosti použití, přesto to je prostředí poměrně neznáme a s nedostatkem vhodných materiálů. Také knihoven s potřebnými funkcemi je poměrně méně. Také z něj vycházející výsledky by nebylo možné rovnou použít v existujících řešeních. Hlavním důvodem zavrhnutí prostředí Matlab je již zmíněná neexistující verze pro nekomerční použití, velmi vysoká cena a tudíž absolutní nedostupnost tohoto prostředí. Jazyk Octave by na řešení téma práce vyhovoval ale výsledky práce by taktéž nebylo možné aplikovat v již existujícím softwarovém produktu. Proto je jazyk Python pro její dokončení nejvhodnější.

Jakožto čistý programovací jazyk je jeho použití víceúčelové a pro jeho použití lze nalézt podstatně více materiálů. Jazyk je v porovnání s ostatními skriptovacími jazyky mnohem efektivnější a rychlejší (ve srovnání například s php). Naprosto svobodná licence pro komerční použití je

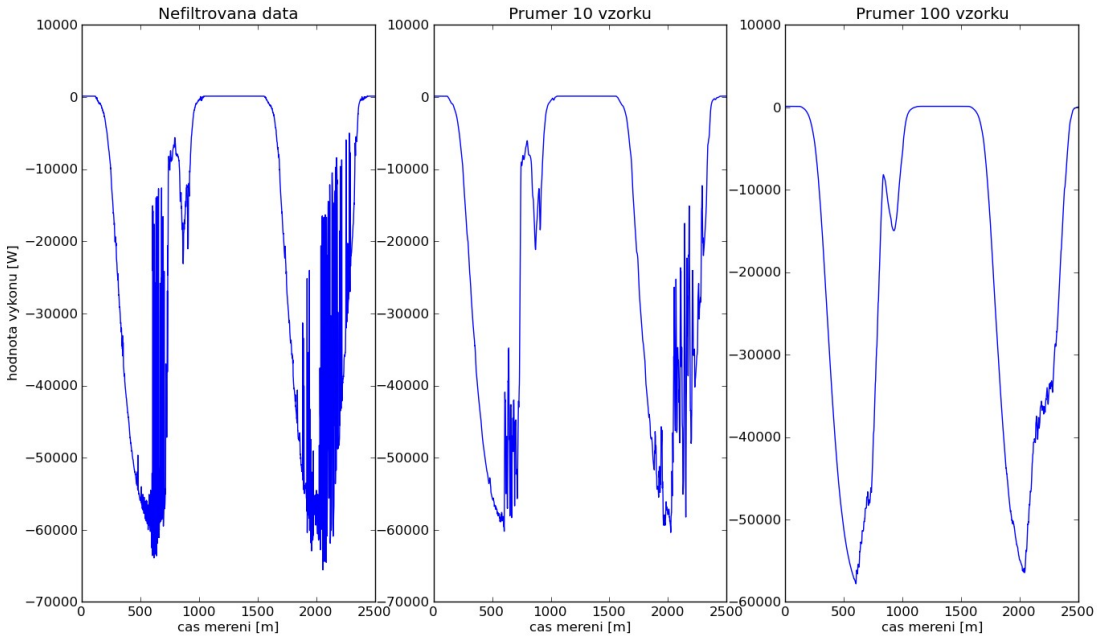
vhodná pro jeho možné budoucí použití v praxi. Díky existenci implementací pro rozšířené objektové jazyky je možné budoucí zakomponování skriptů do již existujících softwarových řešení. Pro Python existují knihovny s funkcemi přímo určenými pro clustering a dobývání znalostí, které se neustále rozvíjejí a jsou vysoce efektivní. Grafická aplikace napsaná v tomto jazyce je skutečně multiplatformní díky vestavěné grafické knihovně GTK v každé implementaci Pythonu. Další velká výhoda jazyka Python je jeho chování v příkazovém interpretu téměř stejné jako Matlab nebo Octave. Můžeme tvořit velké knihovny funkcí a s parametry je volat z interpreta. Tímto způsobem lze vytvořit komplexní nástroj na libovolné operace a používat ho s jednoduchostí příkazové řádky. Vyhneme se tak obsáhlým grafickým prostředím s nepřehlednými volbami a začátečním náporem na uživatele vůbec najít požadovanou akci. Jednou z hlavních předností jazyka je chování a tvoření nových datových typů. Lze v kódu programu vytvářet a plnit složité datové objekty s různými datovými typy (integer, string bool,..) a chovat se k celku jako k poli. V jiném jazyce bychom museli vytvářet komplexní objektové třídy. Ke každému prvku pole lze navíc připojit jeho identifikátor typu string a pomocí něho údaje z pole načítat. Další velmi výhodná vlastnost Pythonu, která je v programovacích jazycích celkem nevídaná, je možnost předávat funkcím jako parametry jiné funkce. Tímto způsobem lze vytvořit například rozsáhlou knihovnu filtrovacích funkcí a podle potřeby je v vkládat. Dokonce lze tyto funkce vkládat do polí, jako jakýkoli jiný datový typ, a odtud je podle potřeby volat, nebo cyklicky provádět. Tato vlastnost je v této práci hojně používaná a při použití jiného nástroje by byla velice složitá. Lze díky tomu jednoduše a rychle kombinovat různé druhy algoritmů pro úpravu dat a následné operace s nimi.

4 Optimalizace záznamu měření

Pro efektivní použití clusterovacích metod je nutné vstupní data před použitím vhodným způsobem zpracovat. Surová data z měření obsahují vysoké množství šumu. Nejprve bylo nutné navrhnout a implementovat postupy, které by tato surová data učinili lépe použitelnými. Pro hledání podobností v souboru je podstatný čistý trend průběhu měření. Reálně naměřená data však obsahují vysoké množství náhodných výkyvů, které nejsou pro zjištění podobnosti podstatné. Proto je nutné data vyhladit či vyfiltrovat některou níže zmíněnou metodou. Dále je pro porovnávání velkého množství souborů nutné omezit počet hodnot pro soubor na efektivní minimum, jinak je průběh algoritmů nesmírně náročný jak na čas, tak i na systémové zdroje. V základním nastavení měří analyzátor SMPQ parametry spotřeby elektrické energie s periodou jedné minuty. Záznam jednoho dne měření tudíž obsahuje 1440 vzorků pro měření jednotlivého parametru, celý soubor jich může obsahovat mnohem víc. Jako příklad je možno uvést měření několika fází výkonu, napětí v čase měření, denní maximum výkonu, atd. Záznam jednoho dne se takto rozroste do velikých rozměrů dat. Proto je nezbytné omezit počet vzorků pro měření, pokud chceme porovnávat podle celkového průběhu. Průběh algoritmu pro určení podobnosti se díky snížení počtu vzorků zefektivní až o jeden časový řád. Další možnost zefektivnění výpočtu je výběr pouze takových vlastností souboru měření, které mají na různorodost jednotlivých vzorků měření vliv. Pokud například každý den měření obsahuje naprosto stejné maximum výkonu, je zavádění tohoto maxima jako parametru pro hledání podobností naprosto irelevantní. V následujících částech této kapitoly budou rozebrány jednotlivé postupy vybrané pro optimalizaci a grafy jejich změn na soubor měření.

4.1 Klouzavý průměr

Filtrace klouzavým průměrem je použita k vyhlazování šumu, náhodných výkyvů měření a odhalování čistého průběhu časové řady. Počítá se jako aritmetický průměr jednotlivé hodnoty a předem určeného počtu hodnot následujících. Dále může algoritmus probíhat tak, že se počítá s požadovaným množstvím předchozích vzorků, nebo se interval rozdělí na polovinu a aktuální hodnota je v jeho středu. Poté se počítá průměr z tohoto intervalu. V následujícím grafu je ukázán průběh filtrování s různě velkým oknem na soubor dvou dnů měření výkonu solární elektrárny.

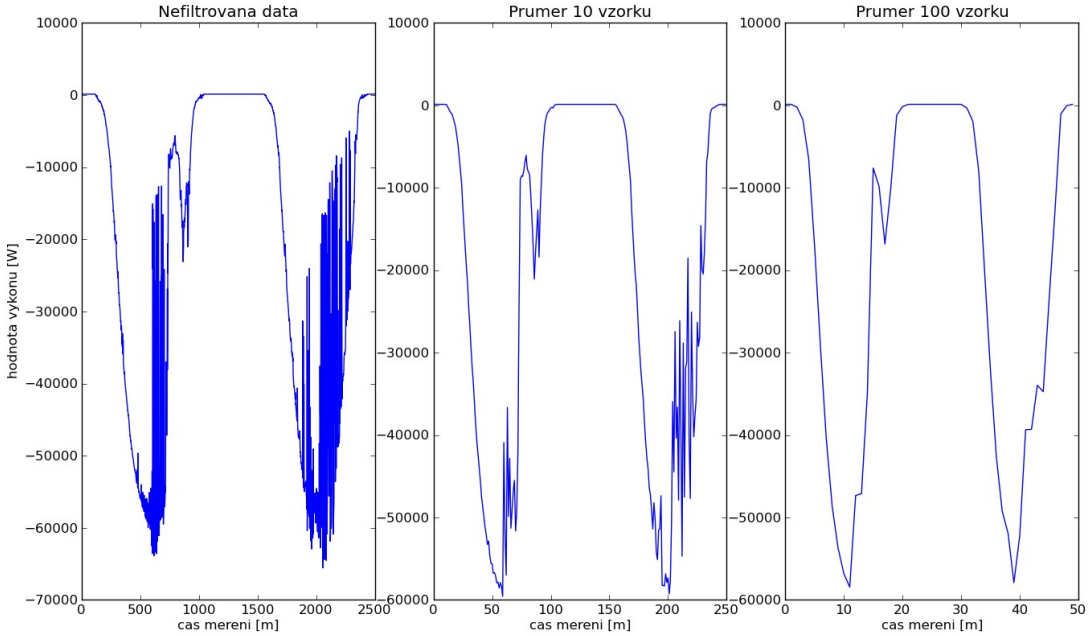


Ilustrace 1: Ukázka filtrace klouzavým průměrem na souboru měření výkonu solární elektrárny

Z grafu jasně vyplývá při použití tohoto filtru je zachován počet vzorků měření ale data lépe a čistěji popisují průběh výkonu během měřeného dne. Výkon elektrárny silně během dne kolísá v závislosti na oblačnosti, a proto je nutné kolísání vyfiltrovat a vybrat ze souboru pouze celkový trend. Algoritmy pro clustering při tomto vyhlazení podávají lepší výkony, protože zbytečný šum ztěžuje vyhledání podobností. Negativní vlastností tohoto algoritmu je poměrně vysoká náročnost jak na čas, tak i na výkon. Musí provést tolik výpočtů průměru, kolik je v souboru dat. Postupně se zvětšováním průměrovacího okna velmi rychle roste doba výpočtu.

4.2 Průměr z „n“ vzorků měření

Při filtraci tímto průměrováním se vypočte a uloží aritmetický průměr pro každých „n“ vzorků v časové řadě. Z toho vyplývá že se počet vzorků zmenší „n“ krát ale pro většinu časových sérií tato metodika dobře popisuje jejich celkový průběh. V následujícím grafu je zobrazeno použití této metodiky na stejný soubor měření jako v předchozím příkladě.



Ilustrace 2: Ukázka filtrace průměrem vzorku na souboru měření výkonu solární elektrárny

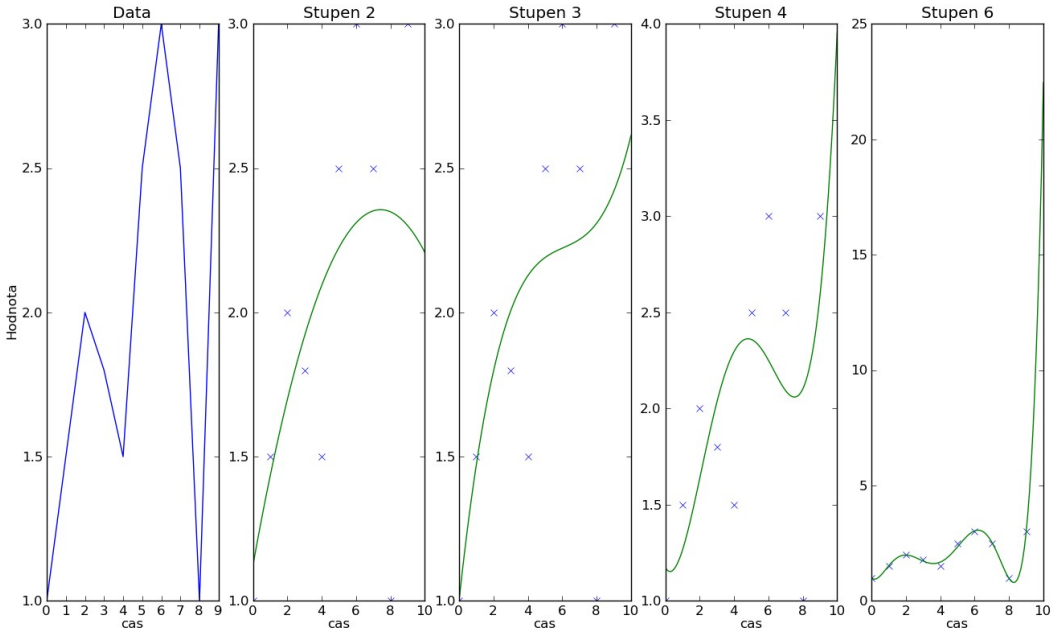
Tato metoda je pro většinu souborů měření nejlepší filtrační technikou. Nejenže v porovnání s klouzavým průměrem podává téměř totožné výsledky, oproti klouzavému průměru na popis trendu měření stačí pouhý zlomek vzorků (klouzavý průměr – 2500, průměr vzorku – 50). Jako další velkou výhodu je nutné uvést mnohem rychlejší průběh výpočtu filtrace než u klouzavého průměru, který je u rozsáhlých souborů měření neméně důležitý. Počet výpočtů průměru je zde oproti předchozí metodě dán vztahem „délka dat/velikost průměrovacího okna“. Podle stejného vztahu se zmenší počet vzorků na den měření. U této metody je nutné dobře zvolit velikost již zmíněného průměrovacího okna. Při příliš vysokém se ztratí podstata dat a jsou pro další porovnávání nepoužitelná. Pro data velkého rozsahu (v řádech desetitisíců vzorků) a při nutnosti zachování relativního šumu lze metodu použít jen pro omezení počtu vzorků. Při velikost

průměrovacího okna 10 se šum zachová téměř v původní podobě ale počet vzorků se sníží o řád.

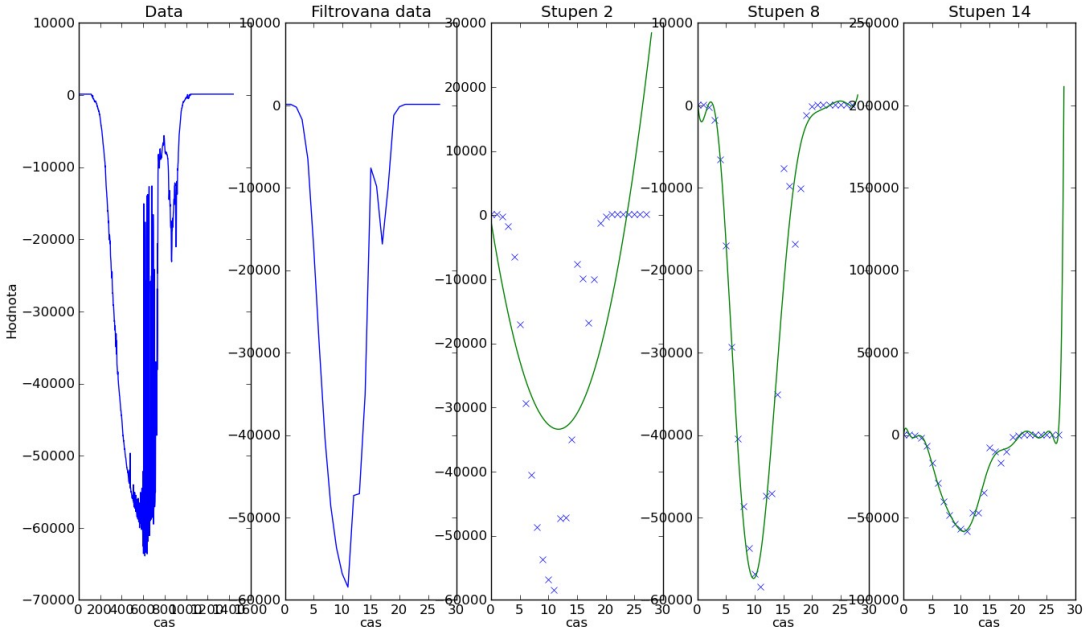
4.3 Polynomiální regrese

Tato metoda je vhodná pouze pro popis trendu malé skupiny dat, ale je ze všech nejefektivnější. Umožňuje popsat velmi malým množstvím parametrů i velice rozsáhlý soubor měření. Algoritmus pro polynomiální regresy se snaží najít polynom předem určeného řádu, který nejlépe „sedí“ na zadaná data. Ideální stav je ,když se podaří dosáhnout stavu, kdy křivka polynomu prochází každým bodem souboru měření. Takového stavu lze ale dosáhnout pouze pro vstupní data malého rozsahu a při použití polynomů velice vysokých řádů (řekněme více než 20. stupně). Takovéto polynomy nejsou vhodné z důvodu vysoké oscilace křivky v částech měření se stagnujícími hodnotami. Proto je vhodné použít polynom v rozmezí 8. až 14. stupně, který sice není ideální ale více méně dobře a efektivně popisuje trend měření. Pro dosažení optimálního výsledku polynomiální regrese je dobré data vyhladit, tudíž vhodně kombinovat s některou z předchozích metodik. Další možností použití je prokládat polynomem celé jednotlivé vzorky, v tomto případě jednotlivé dny měření, nebo vzorek měření rozdělit na části a prokládat polynomem je. Jako příklad je možné uvést rozdělení dnu měření na hodiny a naměřený úsek pro každou hodinu proložit polynomem určeného řádu.

Na následujících grafech budou nejprve ukázány křivky polynomiální regrese několika stupňů na ručně zadaných datech, pro představu fungování algoritmu. Na posledním grafu je ukázka regrese na již dříve použitém souboru měření výkonu solární elektrárny. První podgraf zobrazuje soubor měření vykreslený klasickým způsobem. Druhý podgraf zobrazuje data vyfiltrovaná metodou průměru z „n“ vzorků (v tomto případě 100) pro dosažení optimálních výsledků. Ostatní podgrafy zobrazují průběhy polynomiální regrese, kde nespojitými křížky jsou zobrazeny jednotlivé vzorky měření a spojitou křivkou je zobrazen polynom určeného stupně. Pro tuto metodu je vhodné zvolit právě předfiltrování některou metodou snižující počet vzorků. Vysoké množství šumu v souboru by zbytečně prodlužovalo výpočet polynomu a na výsledné křivce by se jejich přítomnost nijak neprojevila.



Ilustrace 3: Ukázka polynomiální regrese na ručně psaných datech



Ilustrace 4: Ukázka polynomiální regrese na souboru měření výkonu solární elektrárny

4.4 Porovnání optimalizačních metod

Nelze přímo určit, která z popsaných metod je nejefektivnější nebo nejlepší, protože každá podává výsledky o něco jiného druhu a je vhodnější k použití na jiném typu dat. Obecně lze však říci, že metoda pro průměr z „n“ vzorků podává mnohem lepší výsledky co se týče výkonu i efektivity než klouzavý průměr pro použití v clusterovacích algoritmech. Samozřejmě záleží na vhodném zvolení parametrů výpočtu. Pokud nám jde čistě o vyfiltrování šumu a chceme zachovat celá data, je klouzavý průměr mnohem vhodnější.

V následujících tabulkách je uvedeno srovnání efektivity vybraných filtračních algoritmů. Měření je provedeno podle vlastního jednoduchého benchmarku, jehož kód je uveden v příloze. Algoritmus měří čas systémový čas nutný pro výpočet filtrovacího algoritmu. Aby byli rozdíly rychlosti filtračních algoritmů patrné, je použit soubor měření velkého rozsahu a to 10000000 vzorků. Budou použita data z měření výkonu solární elektrárny. Pro algoritmy pro klouzavý průměr a průměr z „n“ vzorků budou použity parametry průměrovacího okna 10, 100 vzorků a měření je uvedeno ve vlastní tabulce. Pro testování rychlosti polynomiální regrese je použita nejprve ukázka rychlosti průběhu na nefiltrovaných datech, poté na datech filtrovaných klouzavým průměrem (průměr 10 vzorků) a jako poslední filtrovaných průměrem z „n“ vzorků (10 vzorků). Jako parametry budou zvoleny stupeň polynomu druhého, pátého, osmého a dvanáctého řádu. Použitá data budou mít stejný rozsah jako v předchozím případě. V poslední tabulce budou uvedeny rychlosti výpočtu filtrace všemi uvedenými algoritmy a následného průběhu algoritmu K-Means a algoritmu pro hierarchický clustering. Parametry a typy filtrace budou použity stejné jako v předchozím příkladě, pro algoritmus K-Means budou použity jako parametry měření vzdálenosti pomocí Pearsonova korelačního koeficientu a clusterování dat do deseti clusterů, rozsah dat je snížen pouze na rozsah třiceti dní. Důvod tohoto měření je porovnání rychlosti výpočtu clusteringu na nefiltrovaných a filtrovaných datech. V případě, že by průběh na nefiltrovaných datech vykazoval lepší výsledky než na filtrovaných, je použití filtrovací metody irelevantní. Protože algoritmus K-Means pro svoji funkci dosazuje na začátku výpočtu náhodné parametry, doba výpočtu polohy clusterů se při každém spuštění o malé množství času liší (výsledky jsou téměř stejné). Proto je pro každé z měření při použití algoritmu K-Means provedeno třikrát a časové výsledky jsou zprůměrovány aritmetickým průměrem.

	10 vzorků	100 vzorků
Klouzavý průměr	149 s	1299 s
Průměr z „n“ vzorků	4 s	4 s

Tabulka 2: Porovnání filtrace klouzavým průměrem a průměrem z "n" vzorků

	Stupeň 2	Stupeň 5	Stupeň 9	Stupeň 12
Nefiltrovaná data	4 s	9 s	16 s	51 s
Klouzavý průměr 10 vzorků	3 s	7 s	14 s	20 s
Průměr vzorku 10 vzorků	0 s	1 s	2 s	2 s

Tabulka 3: Porovnání rychlosti polynomiální regrese v závislosti na stupni polynomu a filtrování dat

	Nefiltrovaná data	Klouzavý průměr 10 vzorků	Klouzavý průměr 100 vzorků	Průměr 10 vzorků	Průměr 100 vzorků
K-Means	8 s	16 s	40 s	2 s	1 s
Hieararch. clustering	9 s	10 s	13 s	1 s	1 s

Tabulka 4: Porovnání rychlosti výpočtu clusterovacích algoritmů v závislosti na filtraci dat včetně času výpočtu filtrace

	Nefiltrovaná data	Stupeň 2	Stupeň 5	Stupeň 9	Stupeň 12
K-Means	8 s	1 s	1 s	1 s	1 s
Hieararchický clustering	9 s	1 s	1 s	2 s	2 s

Tabulka 5: Porovnání rychlosti clusterovacích algoritmů v závislosti na stupni polynomiální regrese včetně času výpočtu regrese

Z tabulky číslo 2 jasně vyplývá, že průměr z „n“ vzorků je neporovnatelně rychlejší než klouzavý průměr se stejnými parametry. Dále že průběh tohoto typu filtrace není závislí na velikosti průměrovacího okna. Vždy se pracuje se stejným rozsahem dat, liší se pouze počet operací výpočtu aritmetického průměru v závislosti na velikosti průměrovacího okna, s rostoucí velikostí okna se počet operací snižuje. Vzhledem i k druhé velice pozitivní vlastnosti, a to snížení počtu potřebných vzorků k zachování téměř stejné informace jako u klouzavého průměru, je tato filtrační technika považována za optimální.

Po zhodnocení výsledků efektivity algoritmů z tabulky číslo 3 lze zjistit, že časový průběh výpočtu polynomiální regrese se razantně zvyšuje v závislosti na zvoleném stupni polynomu. Použití filtrace klouzavým průměrem pomůže snížit čas výpočtu, a pro polynomy vyšších stupňů zvyšuje efektivitu, ale samotný výpočet filtrace je neúměrně dlouhý (viz tabulka 2) a proto jeho použití poskytuje mnohem horší výsledky. Použití algoritmu pro průměr z „n“ vzorků vykazuje bezkonkurenční zvýšení efektivity z použitých filtračních technik. I pro rozsah dat těchto rozměrů vykazuje oproti ostatním měřením téměř neměřitelnou rychlost.

Z tabulky číslo čtyři vyplývá, že použití klouzavého průměru jakékoli velikosti okna pro filtraci dat neúměrně prodlužuje délku výpočtu algoritmu. Nicméně výsledky se můžou o něco zlepšit. Použití průměru z „n“ vzorků trvá mnohem kratší dobu ale i celý výpočet algoritmu se razantně zkrátí. Výsledky rozdělení do clusterů budou podobné jako v případě použití klouzavého průměru. Pro srovnání jsou výsledky zařazeny do přílohy.

Po průběhu polynomiální regrese je výpočet algoritmů pro K-Means clustering velice rychlý ale parametry polynomu nepopíší přesně veškeré vlastnosti průběhu měření. Velkou devízou je ale snadnější hledání velkých odchylek od průměru. Graf popisující K-Means s aplikovanou polynomiální regresí na datech se pro ilustraci nachází v příloze. Stejně rychlý průběh platí i pro hierarchický clustering. Je to dáno razantním snížením počtu parametrů pro jeden měření, a to stupeň polynomu + 1. Ukázka průběhu hierarchického clusteringu po aplikaci polynomiální regrese na data se taktéž nachází v příloze

4.5 Výběr vhodných vlastností

Metody pro výběr vhodných vlastností můžou na vhodných datech velice zvýšit výkon algoritmu pro clustering[12]. Každý vzorek dat je reprezentován výčtem svých vlastností a cílem těchto metod je vybrat pouze ty vlastnosti, které budou mít na průběh algoritmu nějaký vliv. Pokud spolu dvě vlastnosti silně korelují, lze jednu z nich vypustit protože nemůže přinést téměř žádné nové informace. Mezi další možnosti hodnocení patří například porovnání míry kolísání hodnot. V případě že tato míra téměř nekolísá a hodnoty jsou shodné, lze tuto vlastnost také vypustit. Pokud porovnáváme spotřebu elektrické energie v průběhu několika dnů a jako jeden z parametrů si zvolíme maximální denní spotřebu, tak v případě přibližné shodnosti všech denních maxim se tato vlastnost na výsledku algoritmu nijak neprojeví. Snížením počtu vlastností se sníží rozměr výpočetního prostoru. Pro výběr vhodných vlastností existují tři skupiny algoritmů lišicí se podle přístupu k hledání vhodných vlastností k vyřazení a závislosti vlastní metody na délku a kvalitu výpočtu. První z nich jsou úplné či „vyčerpávající“ (exhaustive) algoritmy. Data, která používají tyto algoritmy musí být označena, čili každý prvek souboru musí být označkován, do kterého clusteru náleží. Jejich výpočet probíhá tak, že se vytvoří soubor všech možných kombinací vlastností, poté je na každý prvek tohoto souboru aplikován clusterovací algoritmus a výsledky jsou porovnány. Porovnává se množství chybných zařazení prvků oproti jejich označením. Tímto způsobem je poté podle výsledků zvolena vhodná kombinace vlastností. Výhodou je že lze nalézt optimální kombinaci vlastností, ale na úkor obrovské spotřebě času pro výpočet zejména u porovnávání objektů s velkým množstvím vlastností. Také lze algoritmus spustit a nalezne optimální výsledek bez lidského zásahu. Druhý typ algoritmů představují heuristické algoritmy. Jejich výpočet probíhá na základě předem zvolených a člověkem vložených parametrů. Je nutno nastavit podmínku které vlastnosti musí být zahrnuty. Tím se výpočetní prostor několika násobně sníží a nemusí se počítat s kombinacemi vlastností které požadované prvky neobsahují. Výpočet se mnohem zrychlí ale potřebuje předem zvolené parametry a existuje pravděpodobnost že nebudou objeveny další kombinace vlastností, které podmínky sice nesplňují, ale byly by relevantní. Poslední skupinou algoritmů jsou algoritmy vybírající vhodné vlastnosti na základě náhody. Před průběhem algoritmu se předá jako parametr počet iterací algoritmu, kombinace vlastností jsou zvoleny v každé iteraci náhodně. Jako v předchozím případě se razantně sníží časová spotřeba algoritmu nalezení optimálních výsledků je spíše dílem náhody.

5 Algoritmy pro clustering

Všechny algoritmy snažící se o samostatnou strojovou autonomii při prohledávání dat a obecně při rozhodování patří do vědeckého oboru umělé inteligence (dále UI). Pojem UI byl definován poprvé v 50. letech 20. století a od té doby se obor UI velice prohloubil, lépe definoval podstatu svého výzkumu a jeho podobory se oddělili, když s rostoucím rozsahem přerostli ve vlastní obor. Jedním z nich je obor strojového učení (machine learning). Tato disciplína se zabývá výzkumem a vývojem algoritmů umožňujících počítači učit se, chápat zadaná data a vyčítat z nich jejich struktury a informace v nich obsažené. Některé z těchto metod by nebyli pro člověka možné, protože není v našich silách procházet data obrovského rozsahu dané převážně jako soubory čísel a hledat v nich struktury. Dříve zmíněný algoritmus polynomiální regrese lze také považovat za algoritmus patřící do této množiny, pokud se použije na odhalování budoucího trendu podle počátečních dat časové řady. Algoritmy pro strojové učení lze podle jejich vlastností, nutností lidského zásahu a typu vložených dat rozdělit do dvou skupin. První skupinou je typ algoritmů pro učení s dohledem (supervised learning). Algoritmu je předána množina označených vstupních dat. Tímto označením je míněno předem ohodnocené rozdělení do skupin a každý datový vzorek je označen identifikátorem skupiny do které patří. Algoritmus podle podobností mezi členy jedné skupiny vypočte model podobností, který charakterizuje skupinu. Každý další vložený datový vzorek je porovnáván s těmito modely a je označen jako člen skupiny, jejíž charakteristice nejlépe odpovídá. Tato metoda se souhrnně nazývá klasifikace (classification), rozdělujeme data do předem daných tříd (class). Druhou skupinou je typ algoritmů pro učení bez dohledu (unsupervised learning). Tyto algoritmy lze souhrnně popsat jako algoritmy pro clustering[13]. Obecně lze algoritmy pro clustering popsat jako algoritmy snažící se nalézt strukturu mezi množinou nepopsaných dat. Algoritmus rozdělí množinu do skupin (clusterů, podle toho také clustering), kde si členi jedné skupiny jdou maximálně podobní. Bez tohoto postupu se při práci s velkou množinou nepopsaných dat neobejdeme. V principu nezáleží na typu vložených dat, může to být časová série, množina objektů popsány svými vlastnostmi ale i třeba obrázek nebo zvuková stopa. Nicméně tato práce se zabývá pouze clusteringem časových řad, a v některých případech lze časovou řadu popsat jako objet s vlastními charakteristikami, takže také clusteringem objektů. Podle XXX lze tyto algoritmy rozdělit do několika skupin, Jsou to dělící algoritmy (partitioning methods, v jiné literatuře také nazývané algoritmy založené na vzdálenosti, distance-based methods), hierarchické algoritmy (hierarchical methods), algoritmy založené na hustotě (density-based methods), algoritmy založené na „mřížce“ (grid-based methods) a modelově založené algoritmy (model-based methods).

Všechny názvy jsou překládány z anglického jazyka, české ekvivalenty nejsou běžně používané. Vstupem každého algoritmu je vždy soubor dat (v některých případech dodatečné parametry) a výstupem je několik množin clusterů s do nich rozdělenými daty. V základu lze tuto třídu algoritmů rozdělit do dvou skupin podle vlastnosti, může-li jednotlivý vzorek dat být v právě jednom clusteru nebo může být členem clusterů více. Jako příklad můžeme uvést několik algoritmů ze skupiny dělících algoritmů. Již několikrát zmíněný algoritmus K-Mans je typickým členem skupiny metod, kde jeden vzorek dat může být pouze v jednom clusteru a jeho ekvivalent pracující na stejném principu C-Means, u něhož může být vzorek dat členem vícero clusterů.

V následující části této kapitoly budou podrobně rozebrány jednotlivé algoritmy z hlediska jejich fungování a principů. Budou taktéž prezentovány výsledky práce při použití těchto metod. Je prezentován algoritmus K-Means, jenž byl pro jemu určené parametry určen jako vyhovující a pro řešení daného problému fungující. Dále je popsán algoritmus založený na hierarchických metodách a prezentován výsledek jeho průběhu. Jako poslední prezentováno je několik algoritmů ze skupiny algoritmů založených na vyhodnocování podle hustoty.

5.1 Algoritmy na bázi vzdálenosti prvků

Základním faktorem této skupiny algoritmů je výpočet a porovnávání vzdálenosti mezi jednotlivými prvky souboru. Každý prvek je reprezentován vektorem hodnot, které reprezentují jeho vlastnosti. Jako jednoduchý příklad lze uvést porovnávání ve skupině osob, kde se jako hodnotící parametry zvolí věk, výška a váha. Vektorová reprezentace jednotlivého člověka vypadá například [34,175, 73]. Takovýto vektor lze reprezentovat jako jeden bod v prostoru o rozměrech délky vektoru, v tomto případě v trojrozměrném prostoru. Velikost prostoru není nijak omezena, později je v této práci v některých algoritmech používán až 1440 rozměrný prostor. Dalším krokem k výpočtu algoritmu je vybrání vhodné metriky pro stanovení vzdálenosti dvou bodů. Metriku lze definovat níže uvedenými vlastnostmi.

Mějme množinu bodů, či objektů E a funkci měřící vzdálenost například distance().

Pak platí $\forall\, p,q,r \in E$:

- distance(p, p) = 0
- distance(p, q) ≥ 0
- distance(p, q) = 0 ↔ p = q
- distance(p, q) = distance(q, p)
- distance(p, q) + distance(q, r) ≥ distance(p, r)

Funkce pro měření vzdálenosti musí být definována tak, že pro body, či objekty které si jsou podobné je vzdálenost velice nízká. Níže budou rozebráno několik typů metrik s různými vlastnostmi. Je to Euklidovská vzdálenost, vzdálenost typu Manhatan, Mahalanobisova vzdálenost a Pearsonův korelační koeficient. Díky tomu že v programovacím jazyce Python lze vkládat funkce jako parametry jiným funkcím, je možně předávat clusterovacímu algoritmu jednoduše různě vybrané metriky a zkoumat výsledky. Nelze obecně říci která z těchto metrik je optimální, pro každý typ dat je vhodná jiná a záleží na fázi testování ve které se rozhodne která bude kdy použita. Toto rozhodnutí záleží na lidském faktoru. Nutno dodat že algoritmy na distanční bázi nejsou jediné, které výpočet vzdálenosti v prostoru používají. Například některé z algoritmů na bázi hustoty tyto metody využívají také (například později uvedený DBSCAN). Nicméně pro distančně založené algoritmy je toto měření primární a téměř jedinou metodou.

5.1.1 Euklidovská vzdálenost

Euklidovská vzdálenost je nejjednodušší formou měření vzdálenosti[14]. Vychází z vzorce na měření vzdálenosti bodů z analytické geometrie, který je pouze modifikovanou Pythagorovou větou. Tento typ měření je vhodný pro prostory kde lze měřit vzdálenost mezi body přímo, nebo lze říci vzdušnou čarou.

p, q ∈ E

$$dist(p,q) = \sqrt{\sum_{k=1}^{d}(p_k - q_k)^2}$$

kde d je délka porovnávaného vektoru (počet vlastností objektu)

5.1.2 Vzdálenost Manhattan

Tento typ měření vzdálenosti je vhodný pro prostory kde jsou body rozmístěny přibližně na mřížce a nelze měřit vzdálenost přímo jako u Euklidovské metriky ale pouze po hranách mřížky[15]. Je to stejný typ měření vzdálenosti jako při hledání cesty při průjezdu městem. Silnice představují hrany mřížky a domy její pole, kterým nelze projít (proto název Manhattan, kde jsou budovy
a silnice postaveny téměř jako šachovnice).

p, q ∈ E

$$dist(p,q) = \sum_{k=1}^{d} | p_k - q_k |$$

kde d je délka porovnávaného vektoru (počet vlastností objektu)

Obě tyto metriky lze generalizovat do jednotné formy na takzvanou Minkowského metriku.

p, q ∈ E

$$dist(p,q) = \left(\sum_{k=1}^{d} | p_k - q_k |^r \right)^{\frac{1}{r}}$$

kde d je délka porovnávaného vektoru (počet vlastností objektu), jako parametr r platí :

- 1 = Euklidovská vzdálenost
- 2 = vzdálenost Manhattan

5.1.3 Mahalanobisova vzdálenost

Tuto metriku je vhodné použít všude, kde má na příslušnost k clusteru větší váhu určitá vlastnost než jiné[16]. Je proto nutné jednotlivé vlastnosti objektu normalizovat, neboli přenést na porovnatelnou úroveň. Po normalizaci může každý prvek vektoru nabývat pouze hodnot z intervalu <0 , 1>.

$$dist(p,q) = \sum_{k=1}^{d} \sqrt{\frac{(p_k - q_k)^2}{\sigma_k}}$$

5.1.4 Pearsonův korelační koeficient

Korelační koeficient lze popsat jako míru vzájemného vztahu dvou metrických proměnných[17]. Závislost takovou, že pokud jedna proměnná mění v čase své hodnoty, lineárně je mění proměnná druhá. Tato statistická metoda je pro porovnávání časových sérií použitelná, protože neporovnáváme dva různé nezávislé objekty, ale dvě měření z nichž jedno se odehrálo později než druhé, neboli je to jiná perioda stejného měření průběhu. Proto při jejich vzájemné korelaci nalezneme přibližnou shodu obou měření. Výpočetní vztah lze vyjádřit jako podíl kovariance vektorových proměnných X a Y a násobkem směrodatné odchylky X a směrodatné odchylky Y. Matematicky je vztah vyjádřen na následujícím vzorci.

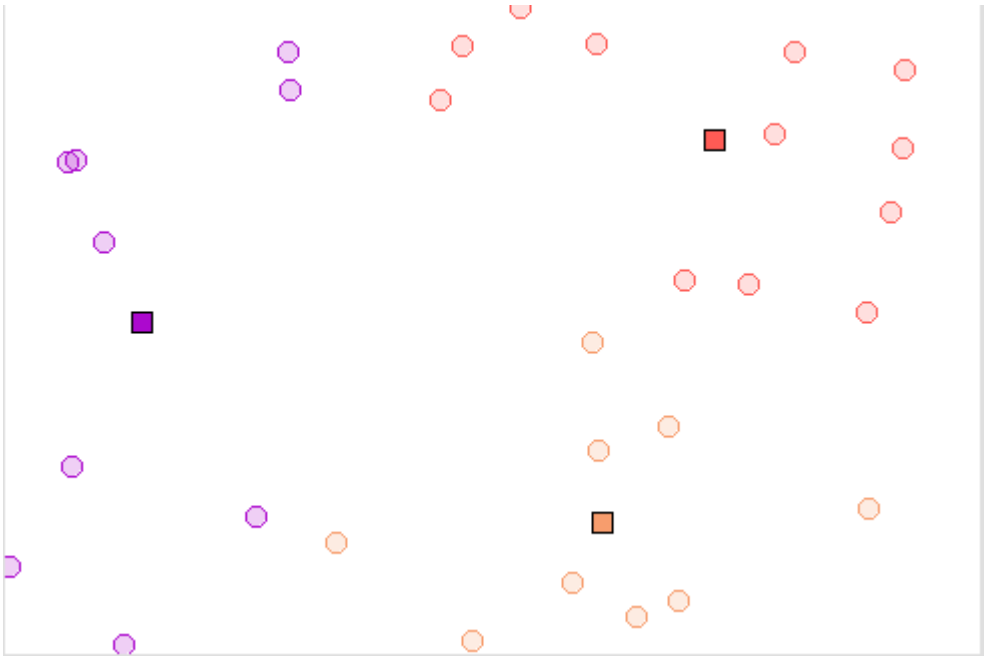
$$r = \frac{s_{xy}}{s_x s_y}$$

Rozšířený výpočetní vzorec aplikovatelný jako algoritmus poté vypadá takto.

$$r = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2 \sum_{i=1}^{n} (y_i - \bar{y})^2}}$$

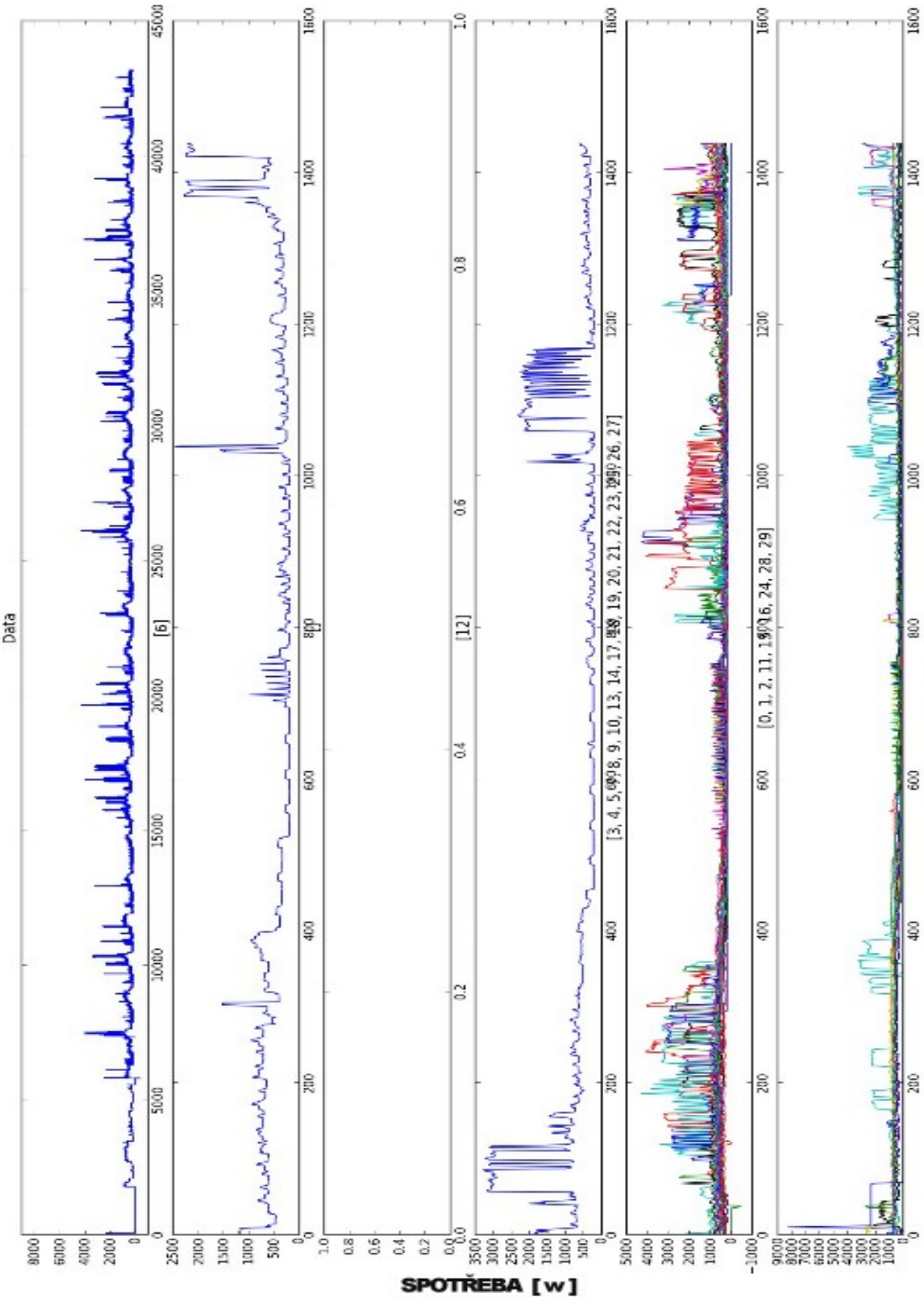
5.1.5 Algoritmus K-Means

Algoritmus K-Means je nejjednodušší ale také velice často používaný clusterovací algoritmus. Funguje na základě měření vzdálenosti mezi objekty v metrickém prostoru, který je definován vybranými vlastnostmi objektů a definovanou měřící funkcí. Tento algoritmus nenajde přesné clustery do kterých jednotlivé objekty patří, ale roztřídí je do předem daného počtu clusterů. Tento počet je nutný optimálně zvolit v závislosti na použitých datech a v nejlepším případě na znalosti přibližného počtu různých typů dat. Které chceme třídit. Jinak se musí na optimální počet přijít testováním. Při použití příliš nízkého počtu clusterů budou v jedné třídě spolu nesouvisející data a také při použití příliš vysokého počtu nastane situace, kdy je několik clusterů zcela prázdných. Algoritmus při inicializaci náhodně zvolí středy jednotlivých clusterů, takzvané centroidy, které se můžou nacházet jak na prázdném poli tak i na některém bodě. Nyní se jednotlivé body zahrnou do toho clusteru, k jehož centroidu mají nejkratší vzdálenost. Nyní se vypočítá průměr pro každou souřadnici výpočetního prostoru pro všechny body náležící k jednotlivému centroidu a vypočte se zprůměrovaná nová poloha centroidu. Toto je konec první iterace a algoritmus iteruje dokud se poloha centroidů neustálí. Ve druhé iteraci by se zase připojily body k jednotlivým centroidům, které jsou k nim blíže díky změně polohy centroidů a opakovalo by se průměrování. Počet iterací závisí na vstupním počtu dat, na rozměrech výpočetního prostoru a na počátečním rozmístění clusterů. Při velkém počtu dimenzí ve vstupních datech trvá výpočet algoritmu velmi dlouhou dobu. Proto je nutné provést na datech optimalizaci některou z dříve jmenovaných metod. Protože se na začátku centroidy inicializují náhodně, je možné že algoritmus podá při opakování o něco jiné výsledky, ale vždy velice podobné. Výsledek algoritmu silně ovlivňují objekty se silně vzdálenými hodnotami, neboli statisticky odlehlá pozorování. Pro řešení této práce to není problém, protože mezi požadované funkce algoritmu patří právě vyhledávání anomálií a odlehlých pozorování, proto lze problém vyřešit vyšším počtem clusterů. Pro představu funkce tohoto algoritmu je možné použít některou z existujících testovacích aplikací. Na následujícím obrázku je ilustrace funkce algoritmu K-Means na náhodných datech. Je zvolen dvourozměrný prostor je objekty jsou reprezentovány dvěma parametry, proto lze vše vykreslit do plochy. Je zvoleno třicet bodů které jsou do plochy rozmístěny a náhodně je zvolena poloha tří centroidů. Obrázek zobrazuje výsledek průběhu algoritmu po třech iteracích. Centroidy jsou znázorněny jako čtverce a body, které náleží do clusteru, který reprezentují jsou kruhy označené stejnou barvou.[18].



Ilustrace 5: Ukázka algoritmu K-Means na náhodných datech

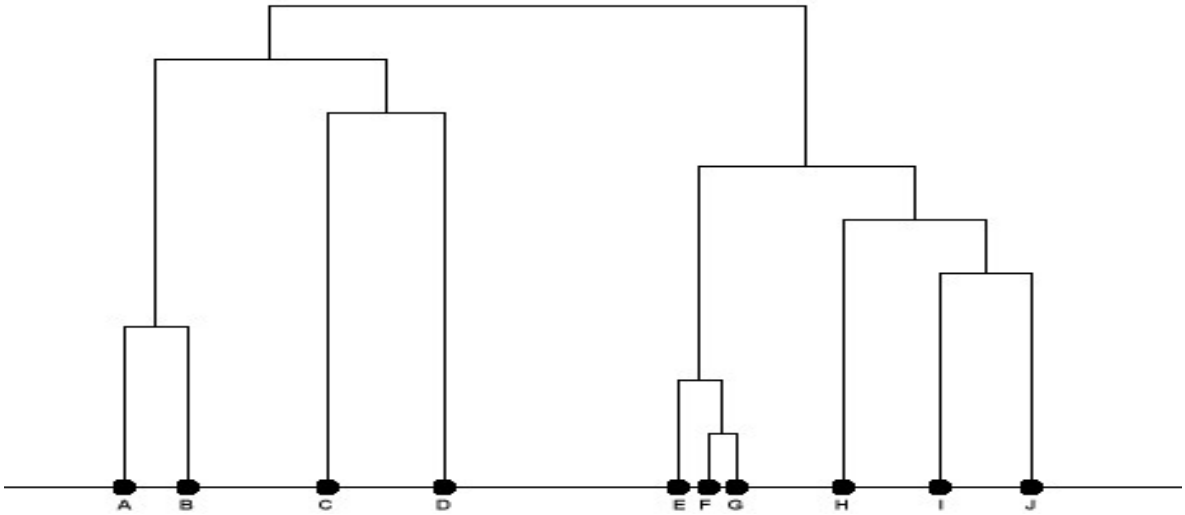
Na ilustraci uvedené níže je ukázka algoritmu použitého přímo na upravená data záznamu parametrů spotřeby elektrické energie. Jedná se o záznam spotřeby obyčejné domácnosti za 30 dnů. Měření je prováděno periodicky každou minutu. Na prvním grafu je pohled na celkový kontinuální záznam za časové období. Tento úsek je poté rozdělen na třicet částí, každá z nich reprezentuje jeden den měření. Zbylých pět představuje jednotlivé clustery a rozdělená data. Vzorky dat jsou pro porovnání naskládané na sebe a každý vzorek je reprezentován vlastní barvou. Identifikátory jednotlivých dnů se nacházejí nad každým podgrafem. Z výsledného grafu jasně vyplývá, že v clusteru číslo čtyři se nachází většina vzorků, které jsou navzájem nejpodobnější. Na pohled lze jasně vyčíst ,že mají průběhy ve stejných okamžicích. Ve zbylých clusterech se nacházejí právě zmíněné anomálie, které je nutné vyhledat. Vzhledem k náhodnému základnímu rozmístění centroidů nelze ovlivnit vyplnění jednotlivých clusterů. Na tomto případě je vidět, že cluster číslo dvě zůstal prázdný. Tomuto jezu lze zabránit snížením počtu clusterů nebo opakovaným spouštěním algoritmu. Vzhledem k prvku náhody je pravděpodobné, že při některém průběhu algoritmu budou všechny clustery obsahovat nějaké vzorky dat. V příloze je také několik grafů pro srovnání výsledků v závislosti na filtraci.



Ilustrace 6: Ukázka algoritmu K-Means na měření spotřeby obyčejné domácnosti

5.2 Algoritmy pro clustering na hierarchické bázi

Algoritmy pro hierarchický clustering tvoří zvláštní škálu, protože jejich výsledky jsou odlišné než u ostatních clusterovacích algoritmů. Výsledkem není množina clusterů a objekty v nich obsažené ale dendrogram zobrazující blízkost jednotlivých objektů. Také tyto algoritmy používají jako svůj základ metrický prostor pro měření vzdálenosti. Jako v předchozím případě je zde každý objekt reprezentován bodem v prostoru a je měřena vzdálenost mezi nimi. Algoritmus poté spojuje nejbližší body a nahrazuje je jedním společným, jehož parametry se získají vypočítáním průměru z obou bodů. Algoritmus tedy nahradí dva nejbližší body jedním a vytvoří tak dva první vrcholy denrogramu. Poté proces opakuje a postupně vytváří tento stromový graf do té doby, než nejsou zahrnuty všechny body. Jako příklad lze použít následující ilustraci, na které se body nacházejí na přímce. Vzdálenost lze tedy posoudit přímo okem.[19]



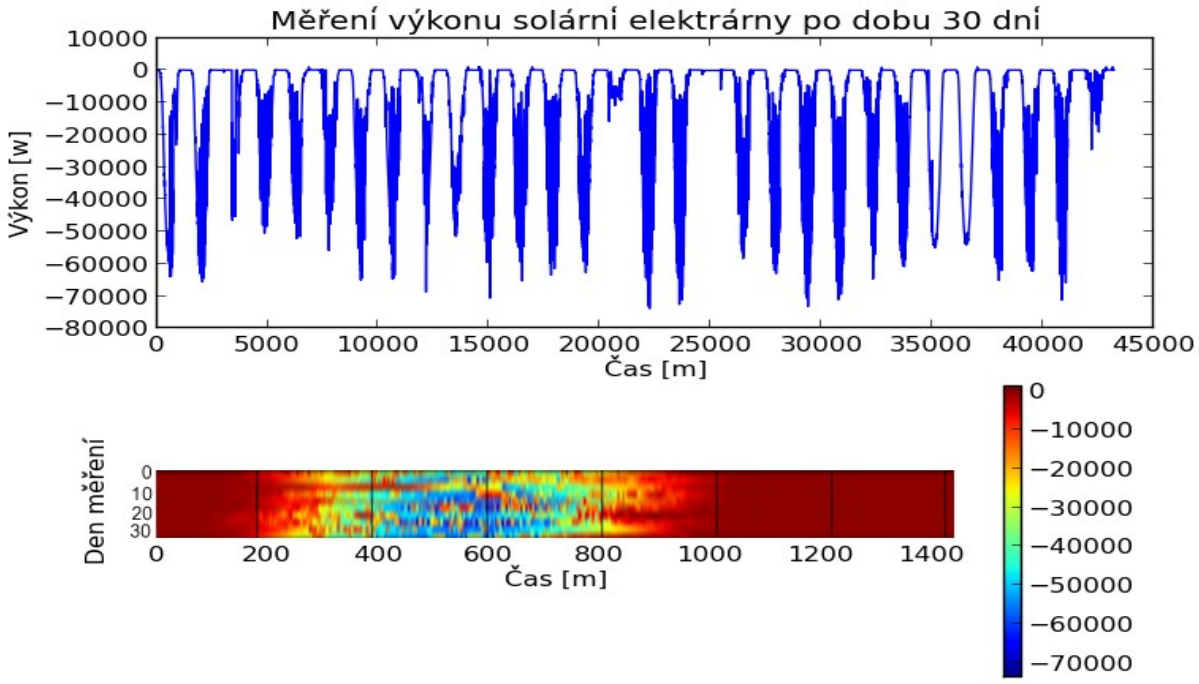
Ilustrace 7: Ukázka hierarchického clusteringu na náhodných datech

V první iteraci algoritmus spojí body F a G, které jsou k sobě nejblíže a vytvoří mezi sebou grafovou vazbu. Vzdálenost mezi nimi se rovná počtu „stupňů“ oddělující jednotlivé části grafu, než dojde k jejich spojení. V tomto případě je tudíž vzdálenost rovna jedné. Ve druhé iteraci se k nim připojí bod E mající následující nejkratší vzdálenost k bodu F. Vzdálenost E od F,G je nyní dvě (oddělují je dva stupně). Algoritmus takto pokračuje dokud nejsou všechny body spojeny do jednoho stromového grafu. Rozdělení souboru do clusterů potom náleží člověku, který výsledek zanalyzuje. V tomto případě je na první pohled vidět, že v souboru se pravděpodobně nacházejí dva clustery, a to tvořeny body [A,B,C,D] a [E,F,G,H,I,J].

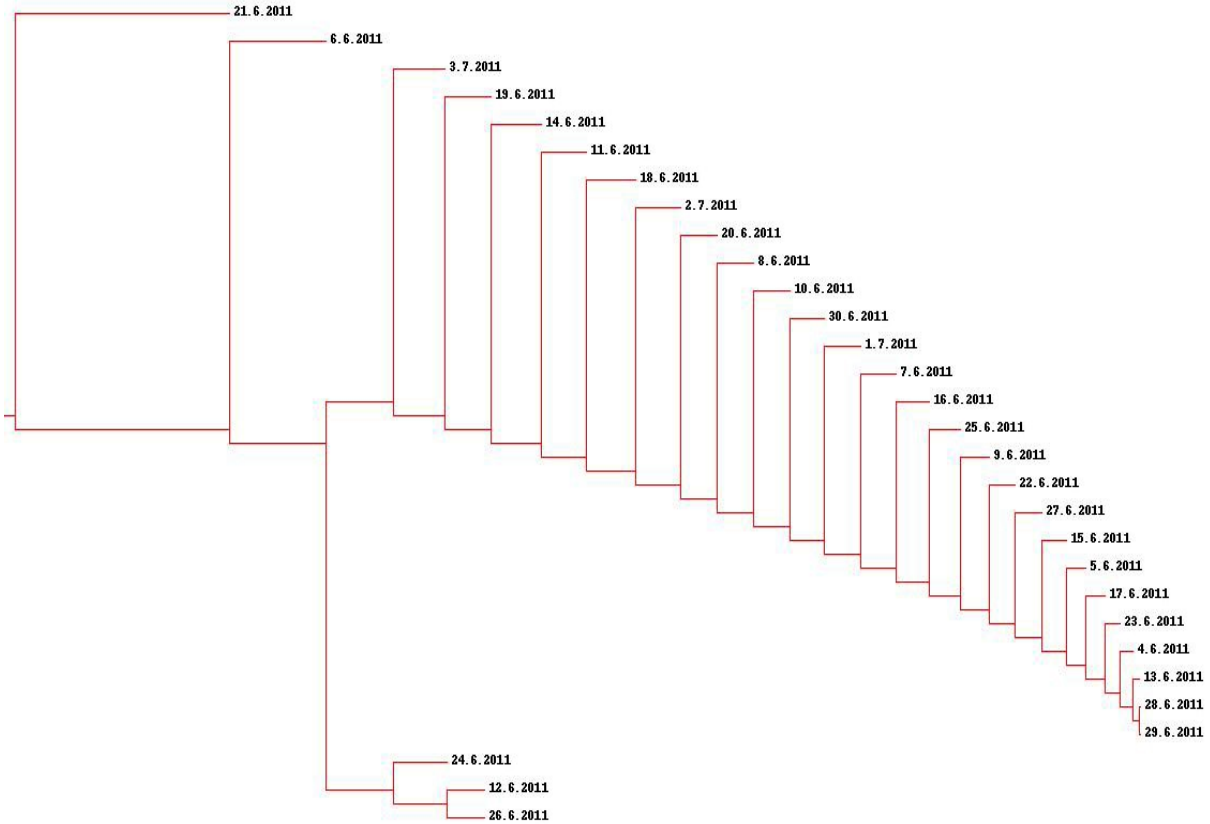
Z ukázkového grafu je patrné že výstupem hierarchických algoritmů není množina clusterů a v nich obsažené objekty, ale prostý stromový graf. O rozložení clusterů neposkytují žádnou informaci a je pouze vizuálního charakteru o vztahu dat. Tato třída algoritmů není příliš vhodná na automatickou analýzu dat, ale je dobrá pro vizuální pohled na systém clusterů v datech a jejich rozložení, nicméně výsledky se dále špatně zpracovávají. Pokud by uživatel neměl absolutně žádnou představu a rozložení dat v souboru, může si takto utvořit představu a poté zkusit některý z ostatních typů filtrovacích a clusterovacích technik. Nejlépe vychází kombinace hierarchického clusteringu a algoritmu K-Means. V první fázi proběhne zjištěný o vizuální podobě dat rozhodnutí o přibližném počtu clusterů, které se poté použijí jako parametry pro K-Means. Pro srovnání rychlosti průběhu je v následující tabulce uvedena závislost rychlosti algoritmu na rozsahu dat a použití některé z filtračních technik. Parametry filtrace jsou podobné jako v předchozím měření, to jest data měření výkonu solární elektrárny o rozsahu 30 dní. Je měřena délka průběhu samotného algoritmu, délka průběhu včetně času výpočtu filtrace je v tabulkách číslo 4 a 5. Poté jsou prezentovány dva grafy srovnávající výsledky hierarchického clusteringu a algoritmu K-Means na datech z 30 dnů měření výkonu solární elektrárny.

	Nefiltrovaná data	Klouzavý průměr 10 vzorků	Klouzavý průměr 100 vzorků	Průměr 10 vzorků	Průměr 100 vzorků
Hierarch. algoritmus	9 s	8 s	8s	2 s	1 s

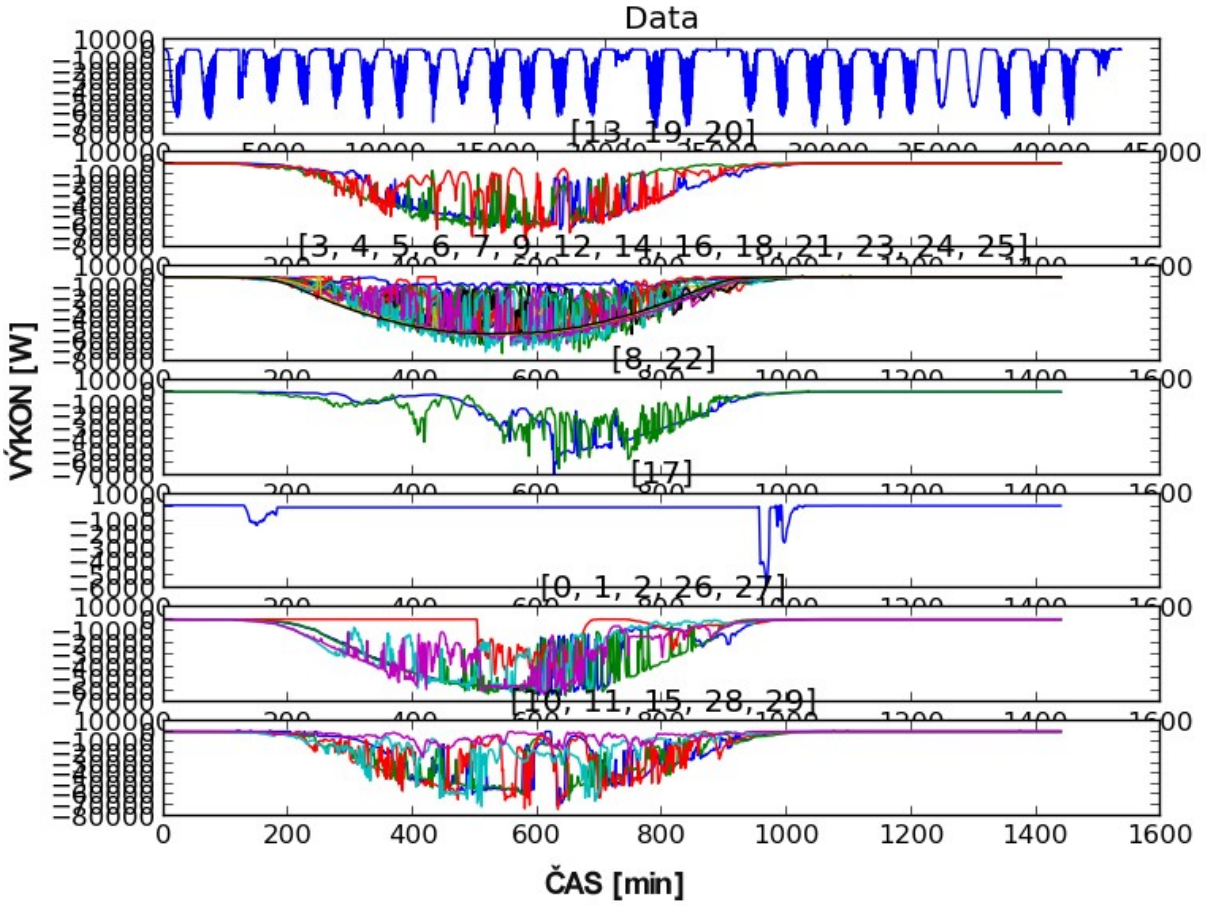
Tabulka 6: Rychlost průběhu samotného hierarchického algoritmu v závislosti na typu filtrace



Ilustrace 8: Ukázka průběhu 30 dnů měření výkonu solární elektrárny



Ilustrace 9: Ukázka výsledku hierarchického clusteringu na 30 dnech měření výkonu solární elektrárny



Ilustrace 10: Ukázka výsledku algoritmu K-Means na 30 dnech měření výkonu solární elektrárny

Pro porovnání výsledků hierarchických algoritmů a algoritmů na bázi vzdálenosti jsou vybrány již několikrát prezentovaná data měření výkonu solární elektrárny. Na prvním grafu v ilustraci číslo osm je vidět čistý časový průběh výkonu po dobu třiceti dní. Je to prosté zobrazení, kde osa Y představuje hodnotu a osa X časový údaj měření v minutách. Na druhém grafu se nachází trojrozměrné zobrazení průběhu. Osa Y představuje jednotlivé dny měření a osa X představuje jejich časové průběhy. Délka osy Y je 30 – třicet dní a délka osy X je 1440, je měřeno po doby 24 hodin každou minutu, tudíž je to 24 * 60 vzorků. Barva každého jednotlivého vzorku značí jeho hodnotu přeložitelnou podle legendy vedle grafu. Na tomto vybraném úseku z prvního grafu jsou na první pohled vidět závažnější odchylky, a to zejména ve dnech číslo 14 a 17, kde data téměř chybí. To může být důsledkem buď celodenního silného zatažení, nebo možného výpadku systému. Další výrazné odchylky se nachází na pozicích 24 a 25 projevující se výraznou „dutostí průběhu“. Z důvodu indexace dní pro standardní průběh a pro algoritmus K-Means z důvodu úspory místa od nuly, je v indexování a v normálním datu posun. Datum dne je o čtyři vyšší než jeho index, v případě že den má index 17, je jeho pravé datum 21. . Při pohledu na hierarchický clustering jsou

data s indexy 24 a 25 správně klasifikovány jako nejbližší, a jsou umístěna v úplně první pozici dendrogramu, to jest s daty 28.6 a 29.6. . K těmto dvěma jsou postupně přidávány nejbližší dny měření, protože tvar jejich průběhu je podobný jako u zbytku dnů. V grafu lze ale nalézt několik velice vzdálených větví signalizující vážné odchylky od průměru. V první větvi to jsou dny s daty 26.6., 12.6. a 24.6. , přeloženo na indexy to jsou dny 22, 8, 20. V případě těchto dnů nejsou rozdíli tak patrné, ale při pohledu na den číslo 8 je znát, že je výrazně úzký a špičatý. U dnů 20 a 22 to již téměř nelze okem rozeznat, bylo by potřeba graf podrobně projít na počítači ve velkém zvětšení, ale přesto lze vyčíst jejich špičatost jako v případě 8 a také vyšší hustota výkyvů v těle. Tato vysoká oscilace výkonu je způsobena samotným charakterem elektrárny, kdy výkon razantně a okamžitě omezuje případná oblačnost a změny můžou proběhnout několikrát za minutu podle síly větru. Při posledním bližším pohledu na dendrogram je nutné vzít v potaz dvě nejodlehlejší pozorování, a to měření z 21.6. a 6.6., indexy dnů jsou 17 a 2. Den s indexem 17 je na první pohled den s největší odchylkou, protože nebyl zaznamenán téměř žádný výkon. Při pohledu na den měření číslo 2 je patrná jeho malá výška a úzký profil. Nyní je nutné porovnat výsledky hierarchického clusteringu a algoritmu K-Means. Při pohledu na ilustraci 10 je patrný stejný výstup popisovaný již v ilustraci 6. První ze sedmi podgrafů je stejné prosté zobrazení průběhu jako v ilustraci 8, ostatní podgrafy reprezentují jednotlivé clustery a dny měření do nich náležící. Barvy v každém z podgrafů reprezentují jednotlivé dny měření, čísla nad každým podgrafem značí indexy dnů. Den s největší odchylkou, tj. s indexem 17 je klasifikován do vlastního clusteru. Dny s indexy 8 a 22 klasifikovaná pomocí hierarchiského clustering do vlastní odlehlé větve jsou i zde klasifikovány do vlastního clusteru jako vzájemně velmi podobné. Tyto dny měření byli nejodlišnější a oba klasifikátory je zařadili správně jako velice odlišné. Data nebyla záměrně nijak filtrována a jako parametry pro porovnání se braly kompletní průběhy měření. Z výsledků vyplývá že přes tyto nejhorší možné podmínky nejodlišnější vzorky měření byly správně klasifikovány do vlastních clusterů. Po aplikování filtrů a případné reprezentaci jednotlivých vzorků měření jejich vlastnostmi (např. maximum, průměr, parametry polynomu, v případě grafů s kladnými hodnotami obsah plochy pod grafem,...) by bylo mnohem lepších výsledků. Pro porovnání je v příloze umístěno několik grafů zobrazujících clusterování stejného souboru dat s aplikacemi různých filtrů.

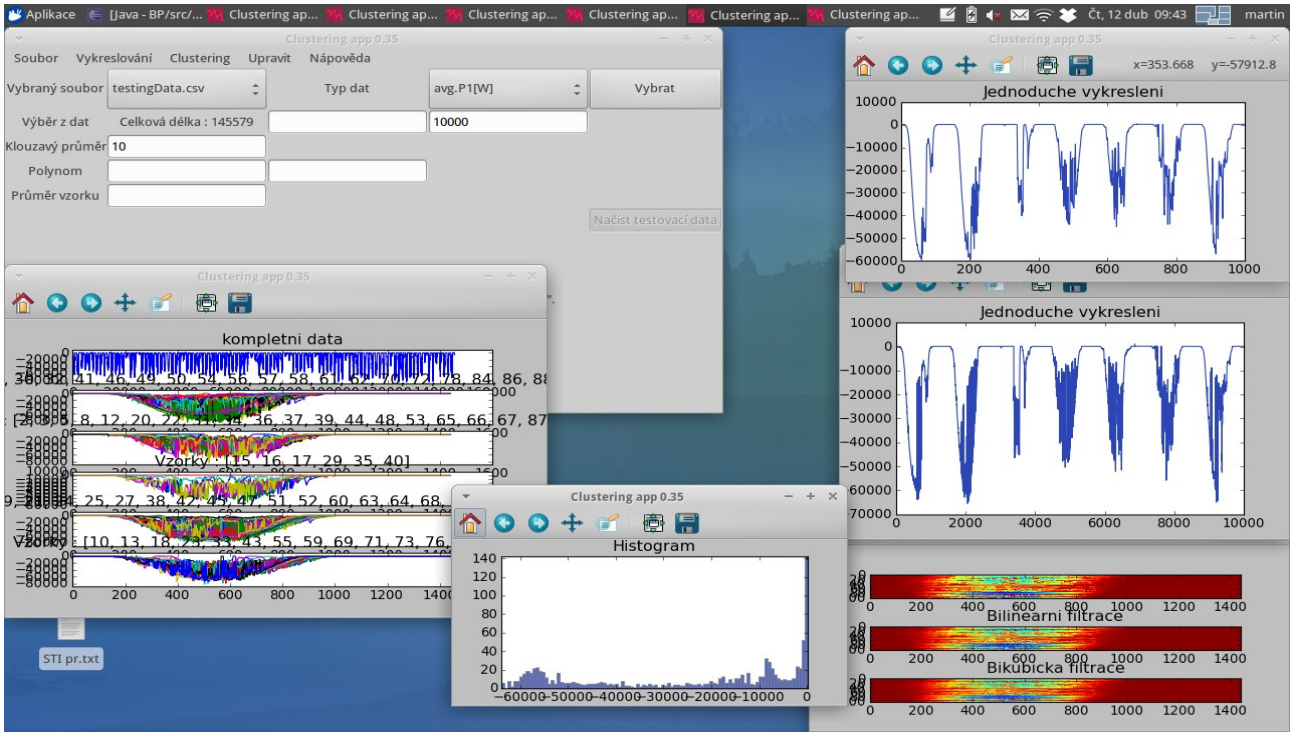
6 Implementace metod

V první fázi samotné implementace bylo testování některých metod zejména v aplikaci R-Project. Po rozhodnutí, že programovací jazyk Python bude nejvhodnější zejména z důvodů jeho možné budoucí integrace do již existujících softwarových řešení, se veškerý vývoj přesunul pouze k jazyku Python. Pro vývoj byla zvolena verze jazyka 2.7 vzhledem ke své majoritní rozšířenosti a na již započatých pracích na interpretu jazyka Python v již zmíněném softwarovém řešení. Celý vývoj probíhal ve vývojovém prostředí Eclipse a výhradně na platformě Linux. V následující fázi byla testována zejména knihovna matplotlib[21] pro vykreslování grafů pro budoucí vizualizaci výsledků různými typy grafů. Poté byly testovány knihovny NumPy[22] a SciPy[23] s nutnými vektorovými a maticovými datovými typy a pokročilými numerickými funkcemi. Byly vytvořeny a otestovány zmíněné filtrační techniky pro pozdější použití. Také bylo nutné vytvořit funkce, které by byly schopné načítat data uložená v databázích. Data bylo možné exportovat v univerzálním formátu csv a ve formátu Microsoft Office Excel xls. Již prezentované algoritmy K-Means a algoritmus pro hierarchický clustering jsou implementované pouze s použitím těchto knihoven. V další fázi byli testovány možnosti knihovny Scikits-Learn[24], obsahující množství již implementovaných algoritmů pro strojové učení a clustering. Několik z těchto algoritmů bylo taktéž otestováno na reálných datech měření. Všechny algoritmy byly ve všech předchozích fázích pouze ve formě samostatně spouštěných skriptů pro samostatné testování.

V poslední fázi byla vytvořena aplikace obsahující a kombinující všechny ukázané metody pro snadné testování a také uživatelsky mnohem přívětivější než samostatné skripty. Aplikace je kompletně naprogramována v jazyce Python s pomocí grafického frameworku PyGTK[25]. Aplikaci je možné spouštět na jakékoli platformě s nainstalovaným jazykem Python a testování aplikace i všech předešlých metod probíhalo jak na platformě Linux tak i na Platformě Windows. Hlavním cílem této aplikace bylo vytvoření nástroje pro snadné testování filtračních technik a algoritmů pro clustering a rychlé pozorování výsledků. Dalším důvodem byla možnost pokračovat na aplikaci v pozdějších pracích, rozšíření jejích možností a možné budoucí využití v praxi.

Aplikace pro clustering umožňuje načtení několika datových souborů najednou, vybírání požadovaného měření z každého z nich (např. výkony, napětí,..). Uživatelské rozhraní je rozděleno do třech částí. První částí je hlavní menu, kde se nachází volby pro načtení souboru, jednoduché vykreslování grafů, menu s volbami pro clusteringové algoritmy a nastavení. Data lze vykreslovat několika způsoby, například již několikrát ukázaným jednoduchým vykreslením, histogramem,

nebo pseudo trojrozměrným zobrazením ukázaném v ilustraci 8. V podmenu pro clustering lze testovat všechny zde prezentované clusteringové algoritmy. Druhou částí je několik prvků pro manipulaci s daty. Nachází se zde dva vysunovací seznamy pro volbu načteného datového souboru (může jich být více na jednou) a pro samotný soubor měření z datového souboru. Načtení dat je vždy nutné potvrdit tlačítkem. Dále jsou zde prvky pro volbu rozsahu dat, se kterými chceme pracovat a prvky pro vložení parametrů filtrování. Zde umožňuje aplikovat filtrační techniky v různých kombinacích, i všechny na jednou a pozorovat výsledky v závislosti na změnách parametrů. Vykreslené výsledné grafy je možné zvětšovat a detailně procházet. Pro zjednodušení práce s programem pro uživatele, který hodlá data testovat, je jeden datový souboru v aplikaci pevně uložen. Je to v této práci již několikrát prezentovaný soubor měření výkonu solární elektrárny. Uživatel si tak může zde prezentované výsledky ověřit v praxi. Pro použití těchto dat je zde přidáno tlačítko, po jehož stisknutí proběhne načtení dat, výběr z jejich parametrů a při prvním použití tak není nutné cokoli nastavovat. Stačí vybrat z menu jednotlivé grafy a nechat je vykreslit. Následující ilustrace prezentuje přehledný snímek obrazovky zobrazující uživatelské rozhraní aplikace a některé z možných grafových výsledků. Na přiloženém cd se nachází tato aplikace spustitelná pomocí interpretu jazyka Python. Také jsou zde veškeré samostatné skripty a tak lze jednotlivé implementace algoritmů spouštět také samostatně.



Ilustrace 11: Ukázka uživatelského rozhraní aplikace pro clustering

7 Závěr

V práci se podařilo splnit všechny body zadání. Proběhlo seznámení se s produkty společnosti KMB sytems, určenými k měření spotřeby elektrické energie. Z mnoha existujících programátorských nástrojů byl vybrán nástroj, s vlastnostmi pravděpodobně nejlepšími pro řešení zadané problematiky. Za největší výhodu vybraného nástroje lze považovat vlastnost, která umožňuje jeho použití jako skriptovacího jazyka do již existujících softwarových řešení. Výsledky této práce tudíž mohou bát dále přímo použitelné. Nástroj je multiplatformní a tak se nelze nijak omezovat jeho budoucím použitím. Jako poslední devízu lze vyzdvihnout jeho otevřenou licenci pro jakékoli, hlavně tedy komerční, řešení. V práci je prezentováno několik filtračních a optimalizačních technik pro práci s daty a redukci jejich velikosti. Všechny výsledky byli otestovány, algoritmy vzájemně porovnány a výsledky jsou přehledně srovnány v tabulkách. Protože filtračních a optimalizačních technik je velké množství, bylo vybráno jen několik z nich. O dalších optimalizačních technikách, které můžou přinést další zlepšení byla napsána alespoň teoretická část. V části zabývající se clusteringem byla vytvořena teoretická rešerže tohoto oboru pro lepší pochopení problematiky. Clustering dat a další dobývání znalostí je taktéž velice široký obor. Proto bylo vybráno pouze několik algoritmů pracujících na naprosto odlišné bázi pro prezentování jejich funkce a výsledků. Každý z algoritmů byl otestován na náhodných datech malého rozsahu, které mohou velice dobře zlepšit pochopení fungování algoritmů. Poté byl prezentován průběh na reálně naměřených datech. Vlastní implementace všech algoritmů proběhla v několika fázích. Algoritmy jsou připraveny na budoucí začlenění do větší knihovny algoritmů pro práci s daty společnosti KMB systems. Výsledkem je také grafická aplikace umožňující v základu uživatelské seznámení s daty a testováním všech prezentovaných algoritmů. Uživateli chápajícímu danou problematiku umožňuje dostávat výsledky algoritmů pro clustering alespoň na základní bázi. Všechny skripty i aplikace jsou připraveny pro budoucí použití a po dalším vylepšení a optimalizaci by se mohli proměnit v plnohodnotný produkt pro clustering záznamu měření parametrů spotřeby elektrické energie.

Seznam použité literatury

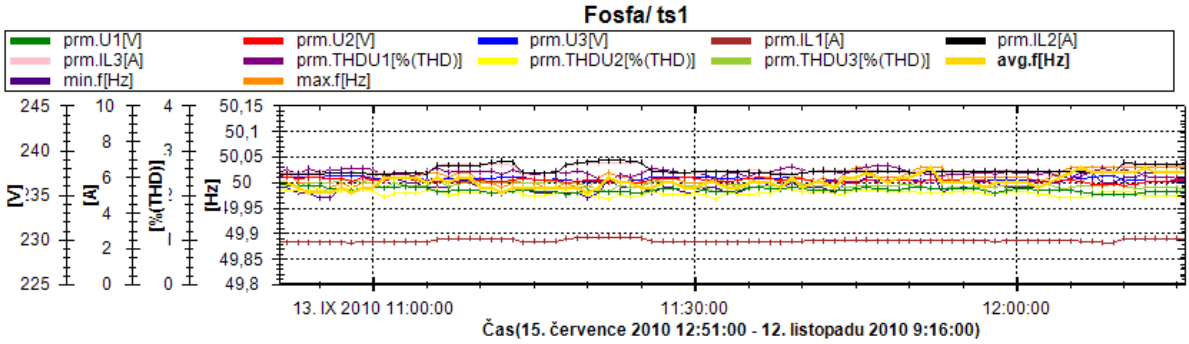
1. KMB systems. *KMB systems* [online]. 2011 [cit. 2012-05-14]. Dostupné z: http://www.kmb.cz/index.php/cs/
2. Harald Hruschka, Martin Natter, Comparing performance of feedforward neural nets and K-means for cluster-based market segmentation, European Journal of Operational Research, Volume 114, Issue 2, 16 April 1999, Pages 346-353, ISSN 0377-2217, 10.1016/S0377-2217(98)00170-2. (http://www.sciencedirect.com/science/article/pii/S0377221798001702)
3. Philip de Chazal; O'Dwyer, M.; Reilly, R.B.; , "Automatic classification of heartbeats using ECG morphology and heartbeat interval features," *Biomedical Engineering, IEEE Transactions on* , vol.51, no.7, pp.1196-1206, July 2004
   doi: 10.1109/TBME.2004.827359
   URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1306572&isnumber=29004
4. Francesco Pattarin, Sandra Paterlini, Tommaso Minerva, Clustering financial time series: an application to mutual funds style analysis, Computational Statistics & Data Analysis, Volume 47, Issue 2, 1 September 2004, Pages 353-372, ISSN 0167-9473, 10.1016/j.csda.2003.11.009. (http://www.sciencedirect.com/science/article/pii/S0167947303002780)
5. Chicco, G.; Napoli, R.; Postolache, P.; Scutariu, M.; Toader, C.; , "Customer characterization options for improving the tariff offer," *Power Systems, IEEE Transactions on* , vol.18, no.1, pp. 381- 387, Feb 2003
   doi: 10.1109/TPWRS.2002.807085
   URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1178823&isnumber=26471
6. Manuál SMV, SMP a SMPQ. In: [online]. [cit. 2012-05-15]. Dostupné z: http://kmb.cz/index.php/cs/merici-pristroje-produkt/smv-smp-a-smpq
7. ENVIS 1.0 - User Guide. In: [online]. [cit. 2012-05-15]. Dostupné z: http://kmb.cz/index.php/cs/aplikace/envis
8. R-Project. [online]. [cit. 2012-05-15]. Dostupné z: http://www.r-project.org/
9. Mathworks Matlab. [online]. [cit. 2012-05-15]. Dostupné z: http://www.mathworks.com/help/techdoc/
10. GNU Octave. [online]. [cit. 2012-05-15]. Dostupné z: https://www.gnu.org/software/octave/
11. Python. [online]. [cit. 2012-05-15]. Dostupné z: http://www.python.org/
12. Dash, M.; Choi, K.; Scheuermann, P.; Huan Liu; , "Feature selection for clustering - a filter solution," *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on* , vol., no., pp. 115- 122, 2002
    doi: 10.1109/ICDM.2002.1183893
    URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1183893&isnumber=26564
13. T. Warren Liao, Clustering of time series data—a survey, Pattern Recognition, Volume 38, Issue 11, November 2005, Pages 1857-1874, ISSN 0031-3203, 10.1016/j.patcog.2005.01.025. (http://www.sciencedirect.com/science/article/pii/S0031320305001305)
14. DEZA, Elena a Michel Marie DEZA. *Encyclopedia of Distances*. Berlin: Springer, 2009. ISBN 978-3-642-

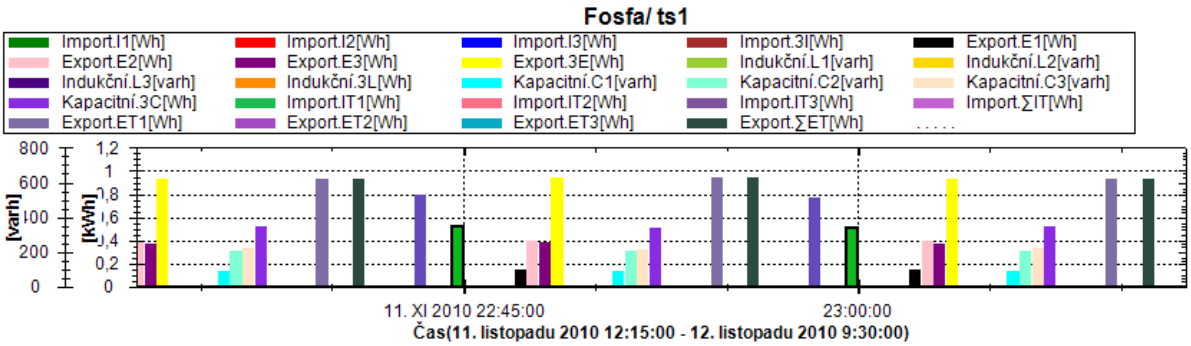
00233-5. Dostupné z: http://www.springerlink.com/content/978-3-642-00233-5/#section=724968&page=1

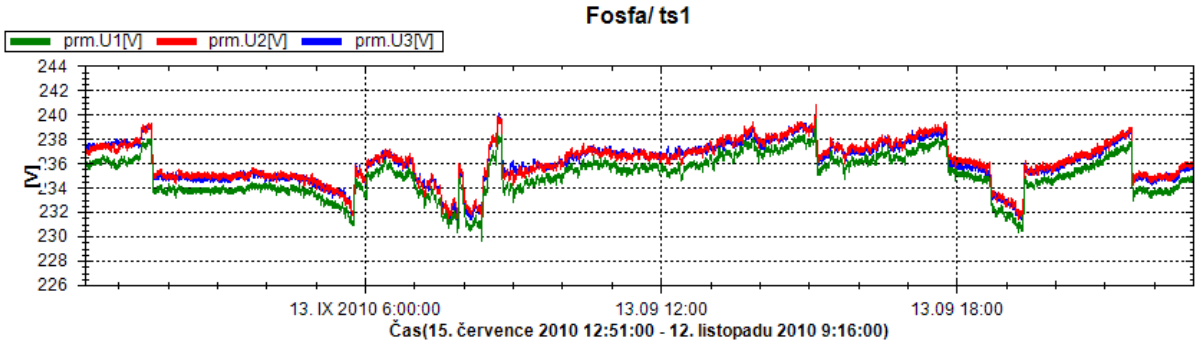
15. GARDNER, Martin. *The Last Recreations*: *Taxicab Geometry*. New York: Springer, 1997, s. 159-175. ISBN 978-0-387-30389-5.
16. Shiming Xiang, Feiping Nie, Changshui Zhang, Learning a Mahalanobis distance metric for data clustering and classification, Pattern Recognition, Volume 41, Issue 12, December 2008, Pages 3600-3612, ISSN 0031-3203, 10.1016/j.patcog.2008.05.018.(http://www.sciencedirect.com/science/article/pii/S0031320308002057)
17. COHEN, Israel, Yiteng HUANG a Jingdong CHEN. *Pearson Correlation Coefficient*: *Noise Reduction in Speech Processing*. Berlin Heidelberg: Springer, 2009, s. 1-4. ISBN 978-3-642-00296-0.
18. K-Means demo. [online]. [cit. 2012-05-15]. Dostupné z: http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html
19. Hierarchical clustering demo. [online]. [cit. 2012-05-15]. Dostupné z: http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletH.html
20. density based clustering
21. Matplotlib. [online]. [cit. 2012-05-15]. Dostupné z: http://matplotlib.sourceforge.net/
22. NumPy. [online]. [cit. 2012-05-15]. Dostupné z: http://numpy.scipy.org/
23. SciPy. [online]. [cit. 2012-05-15]. Dostupné z: http://www.scipy.org/
24. Scikit-Learn. [online]. [cit. 2012-05-15]. Dostupné z: http://scikit-learn.sourceforge.net/stable/
25. PyGTK. [online]. [cit. 2012-05-15]. Dostupné z: http://www.pygtk.org/

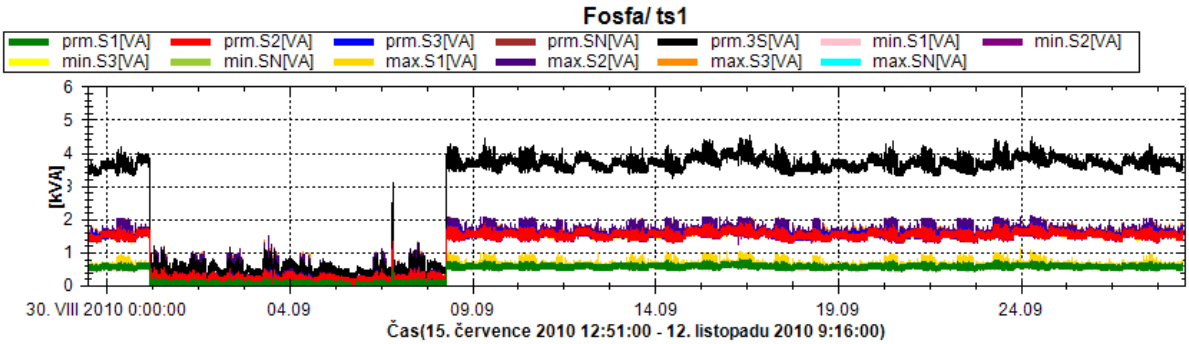
Příloha A – Měřící přístroj SMPQ

Příloha B – Ukázka dat z archivu měřícího přístroje SMPQ

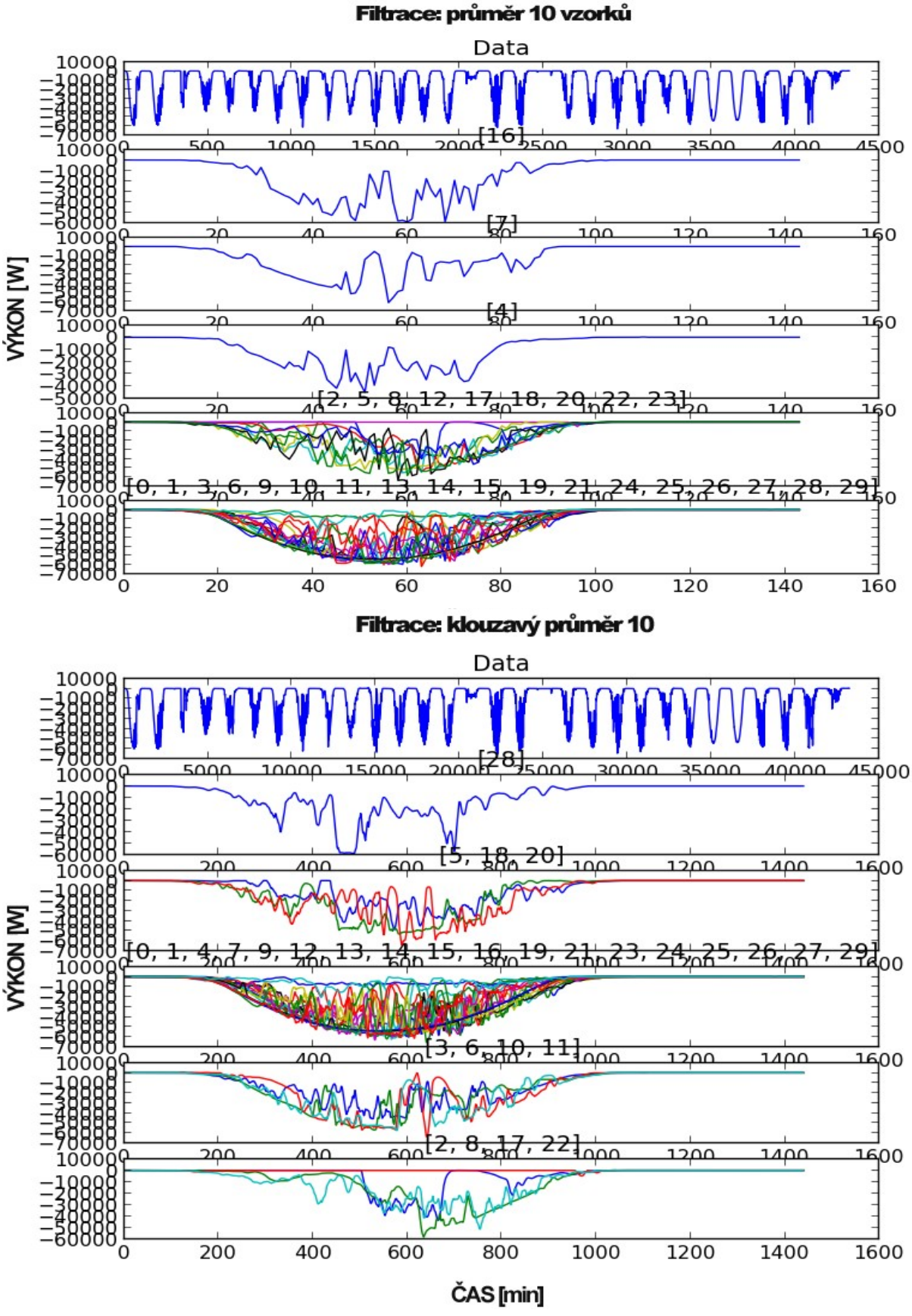


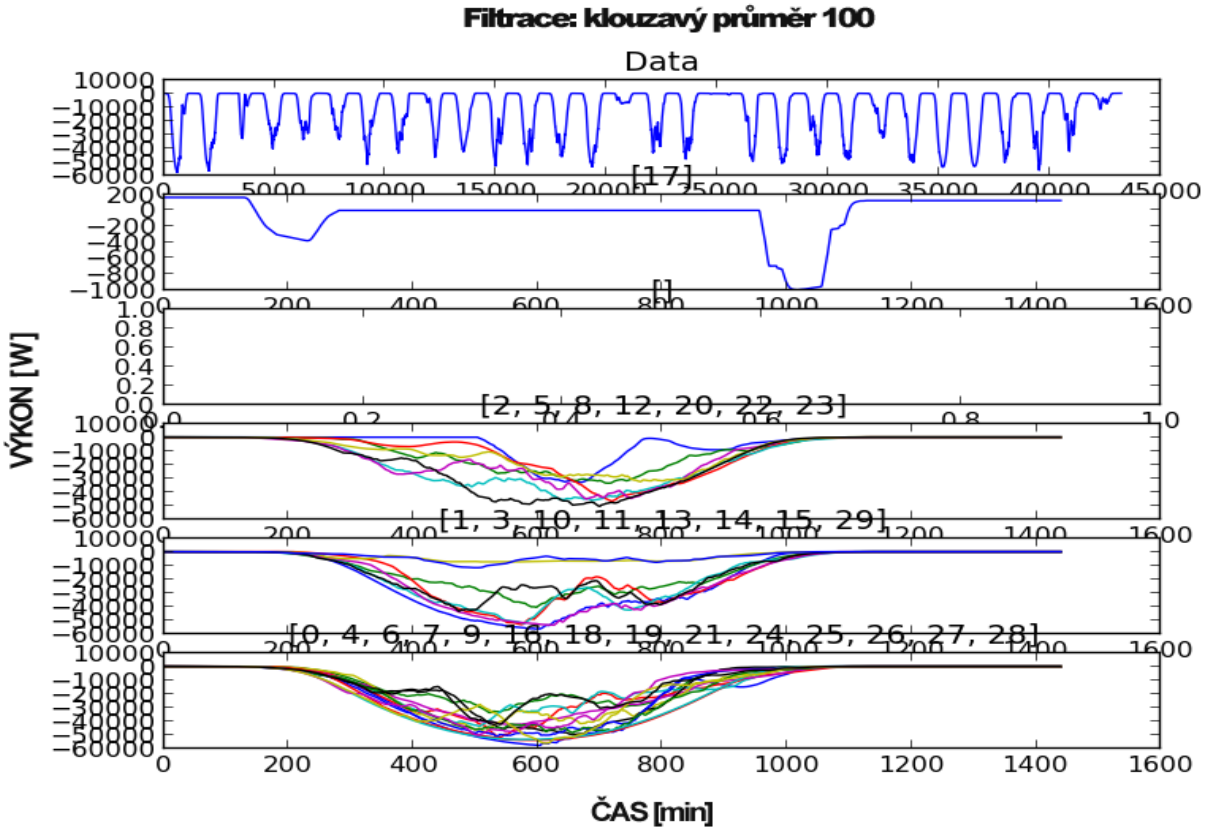


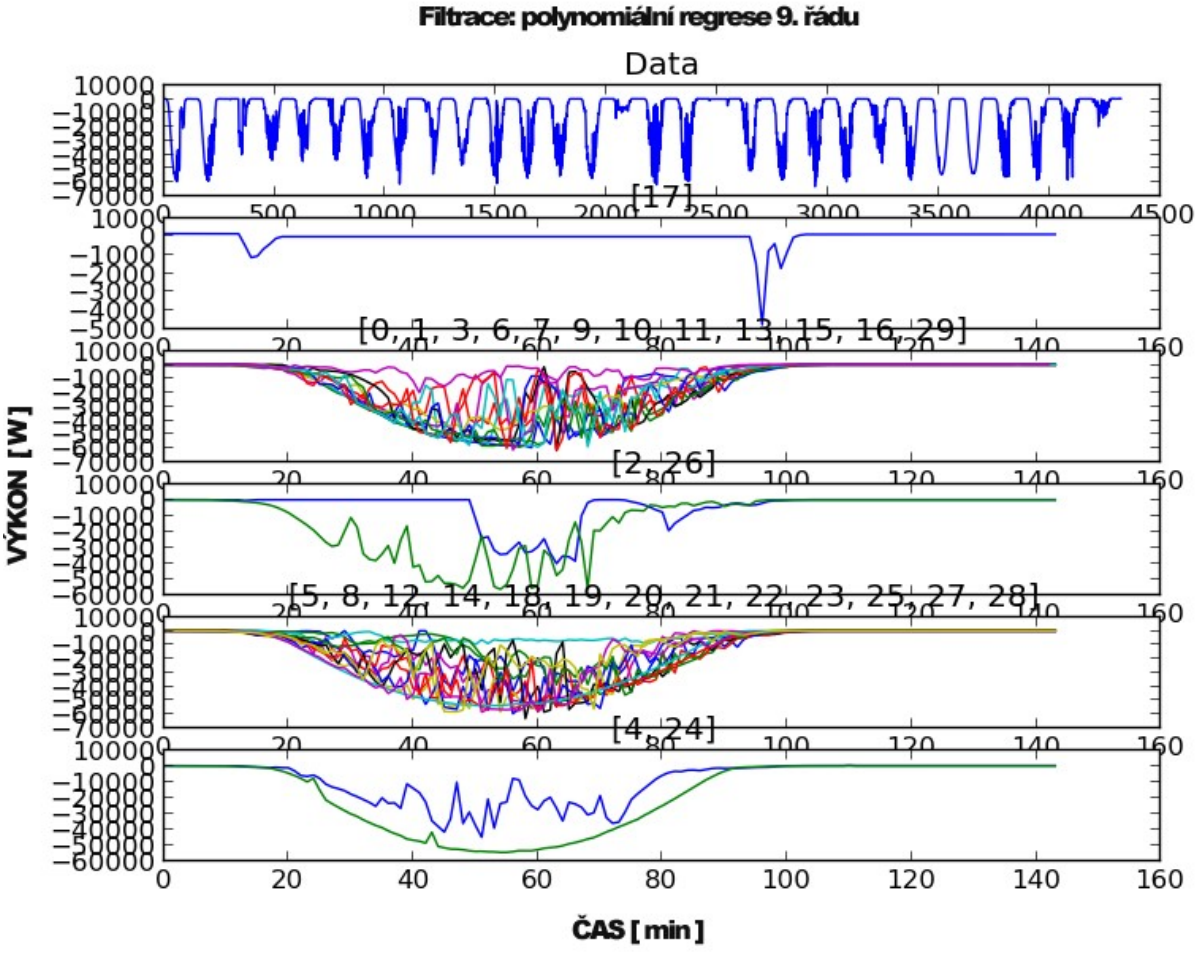




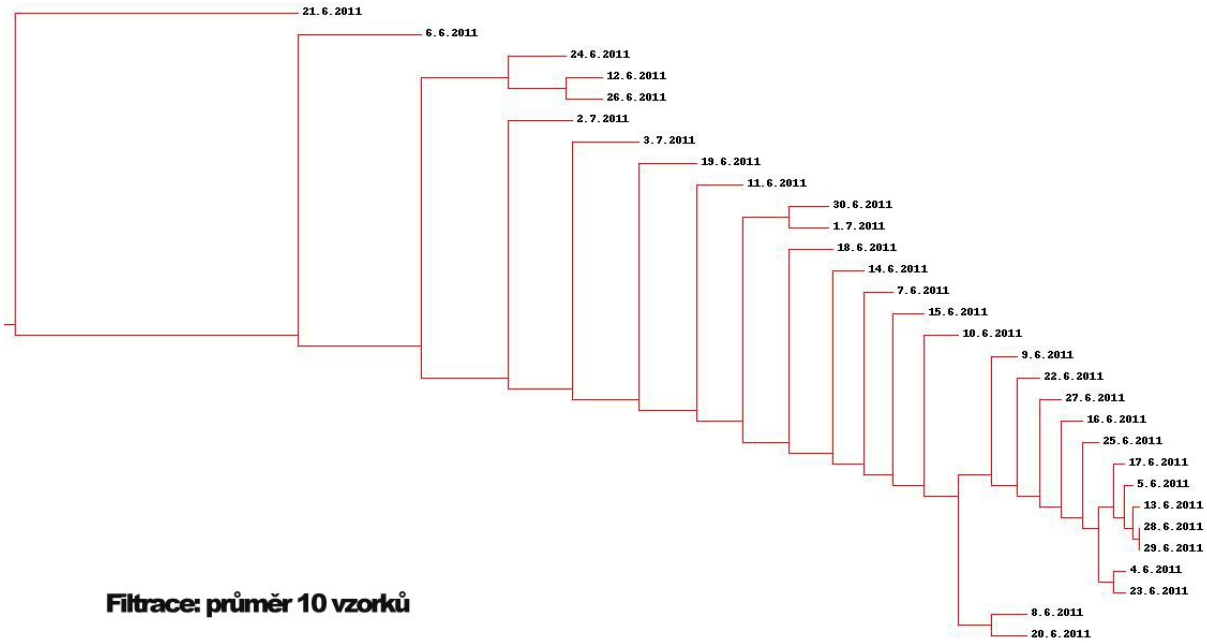
Příloha C – Ukázka výsledků algoritmu K-Means v závislosti na filtraci dat



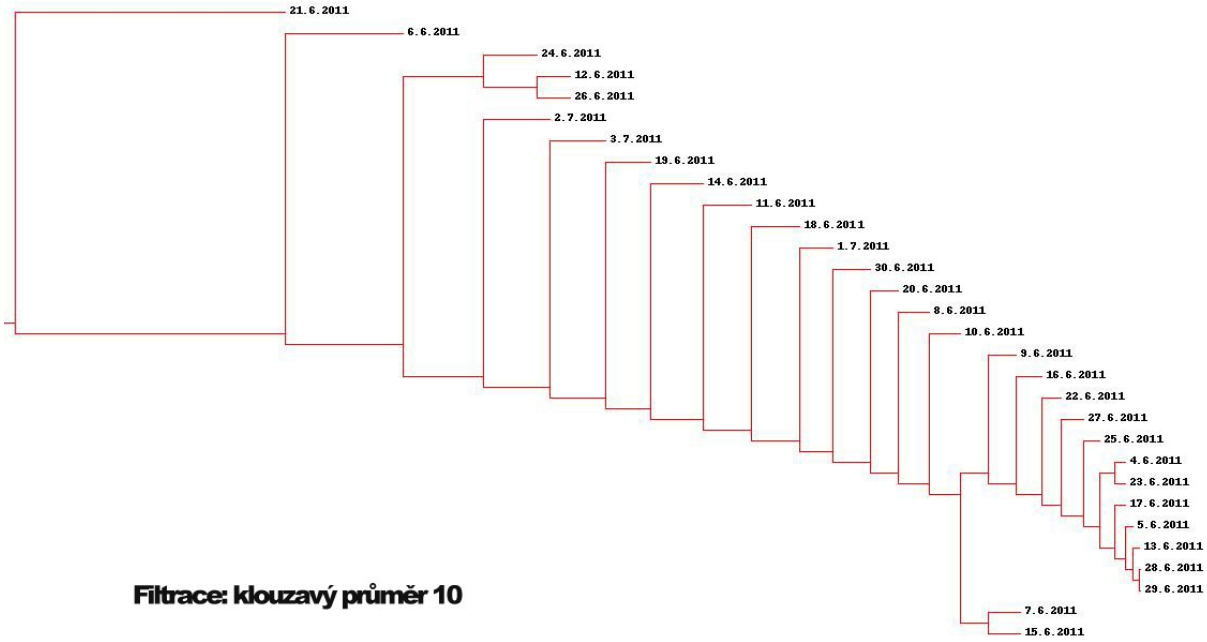

Filtrace: klouzavý průměr 100
Data
[17]
[]
[2, 5, 8, 12, 20, 22, 23]
[1, 3, 10, 11, 13, 14, 15, 29]
[0, 4, 6, 7, 9, 18, 18, 19, 21, 24, 25, 26, 27, 28]
VÝKON [W]
ČAS [min]


Filtrace: polynomiální regrese 9. řádu
Data
[17]
[0, 1, 3, 6, 7, 9, 10, 11, 13, 15, 16, 29]
[2, 26]
[5, 8, 12, 14, 18, 19, 20, 21, 22, 23, 25, 27, 28]
[4, 24]
VÝKON [W]
ČAS [min]

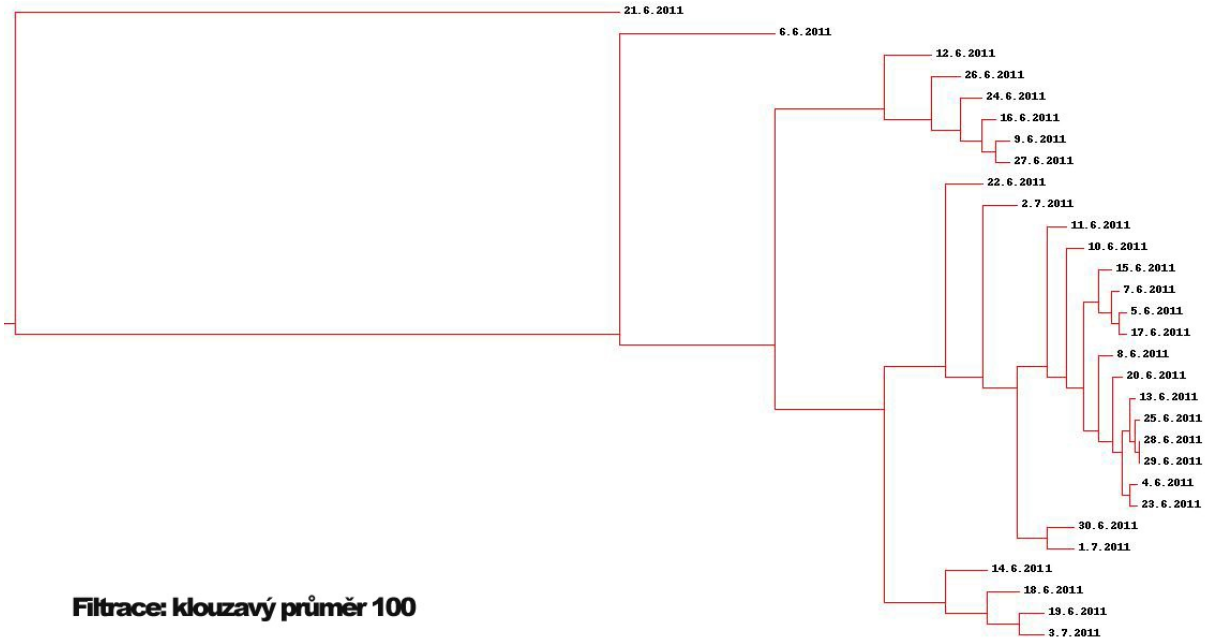
Příloha D – Ukázka výsledků hierarchického clusteringu v závislosti na filtraci dat



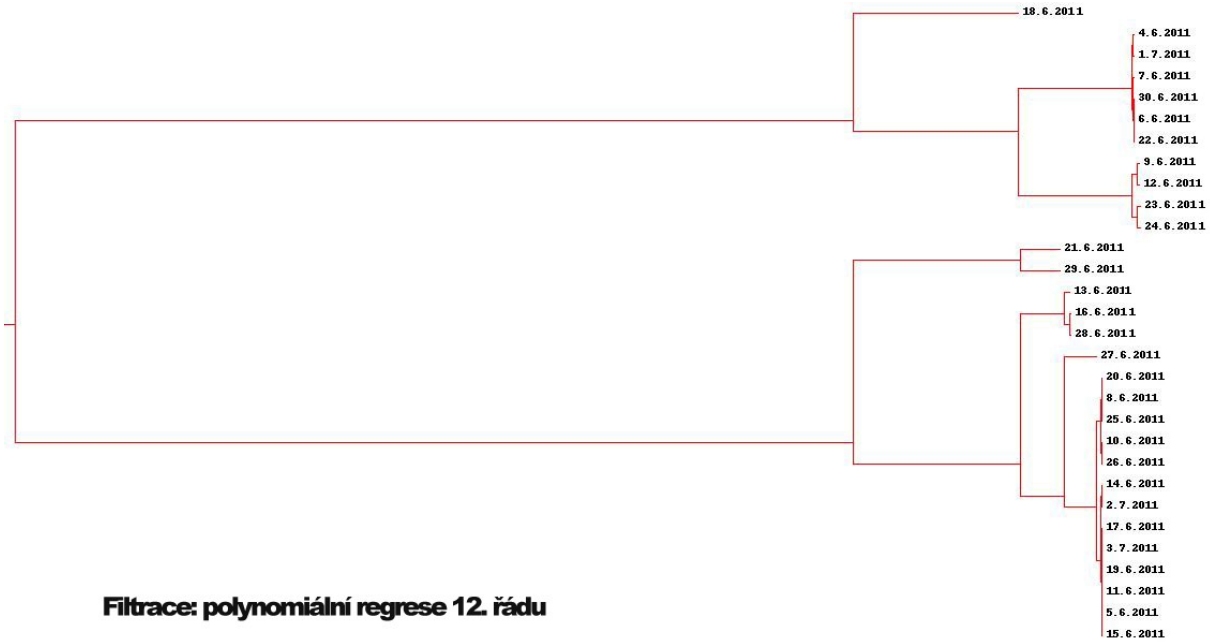
Filtrace: průměr 10 vzorků



Filtrace: klouzavý průměr 10



Filtrace: klouzavý průměr 100



Filtrace: polynomiální regrese 12. řádu